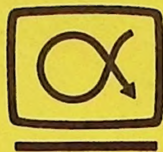
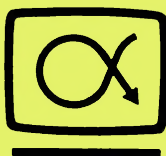


STANISŁAW WALIGÓRSKI

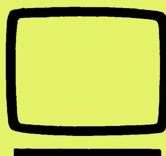
LOGO
NA SINCLAIR SPECTRUM

CZĘŚĆ 1 • dla początkujących

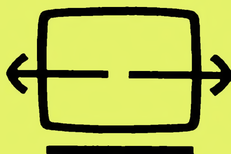
Seria $1_2 + 1_2 = 1Ø_2$ to cykl książek prezentujących zagadnienia informatyki na poziomie maksymalnie dostępnym dla każdego Czytelnika i w zakresie najbardziej mu przydatnym. Poniżej przedstawiony podział serii na grupy tematyczne (każda pozycja cyklu jest oznaczona na okładce odpowiednim do omawianego zagadnienia symbolem grupy) ułatwi Czytelnikowi usystematyzowanie przekazywanych treści oraz zgodny z potrzebą i zainteresowaniami dobór poszczególnych tomików.



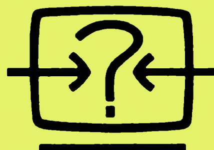
OPROGRAMOWANIE
software



SPRZĘT
hardware



ZASTOSOWANIA KOMPUTERÓW



PORADY DLA MAJSTERKOWICZÓW

STANISŁAW WALIGÓRSKI

LOGO
NA SINCLAIR SPECTRUM

CZĘŚĆ 1 • dla początkujących

INSTYTUT WYDAWNICZY ZWIĄZKÓW ZAWODOWYCH
WARSZAWA 1987

Książka została wydana we współpracy z Redakcją
popularnego miesięcznika matematyczno-fizyczno-astronomicznego „Delta".
Konsultacja z ramienia „Delty" *Zbigniew Odrowąż-Sypniewski*

Projekt graficzny okładki, strony tytułowej,
układ graficzno-typograficzny tekstu
oraz wykonanie ilustracji *Witold Gidzewicz*

Zdjęcia *Adam Płuciennik*

Redaktor Wydawnictwa *Tomasz Majchrowski*

Redaktor techniczny *Barbara Charewicz*

Korektor *Barbara Piekarska*

ISBN 83-202-0522-0

INSTYTUT WYDAWNICZY ZWIĄZKÓW ZAWODOWYCH
WARSZAWA 1987
Wydanie I. Nakład 49.700+300 egz.
Ark. wyd.3,6; ark. druk. 3,5.
Papier offset kl. III, 80 g, 61×86
Oddano do sładania 12 maja 1986 r.
Podpisano do druku w listopadzie 1986 r.
Druk zakończono w styczniu 1987 r.
Nr prod. Ww/1016/KE/85. zam. 270/86. P-47.
DRUKARNIA INSTYTUTU WYDAWNICZEGO
ZWIĄZKÓW ZAWODOWYCH

Wstęp. Czemu Logo?

Logo, najpopularniejszy obecnie spośród języków programowania przeznaczonych do początkowego nauczania w szkołach, zaczyna się rozpowszechniać także u nas. Jest w tej chwili dostępny na wszystkich typach mikrokomputerów używanych w Polsce, z wyjątkiem typów najbardziej prymitywnych lub nieudanych konstrukcyjnie. Wymaga grafiki oraz przynajmniej 48 kilobajtów pamięci o swobodnym dostępie (Random Access Memory - RAM). Ponieważ komputer domowy ma na ogół przynajmniej 16 kilobajtów pamięci stałej (Read Only Memory - ROM), daje to w sumie przynajmniej 64 kilobajty pamięci operacyjnej i tyle rzeczywiście mają najmniejsze mikrokomputery, dostępne w kraju i wyposażone w Logo. Najbardziej popularnymi w kraju mikrokomputerami z Logo są Apple II i różne jego kopie, Commodore 64 i 128, Amstrad CPC 464 i 6128, najrozmaitsze wersje i kopie IBM PC, ale głównie ZX Spectrum 48k firmy Sinclair Research Ltd. Popularność swą Spectrum zawdzięcza przede wszystkim cenie, która, choć mimo pewnego spadku nadal wysoka jak na kieszeń przeciętnego nabywcy a nawet szkoły, może jednak konkurować z cenami innych typów mikrokomputerów.

Główną przeszkodą w rozpowszechnianiu Logo jest brak literatury. Zdobycie dyskietki lub kasetki z językiem jest o wiele łatwiejsze niż zdobycie podręcznika, zwłaszcza takiego, który by pokazywał, jak można się tym językiem skutecznie i wygodnie posługiwać, a jednocześnie zawierał minimum niezbędnych informacji o właściwościach tej wersji języka, którą się ma do dyspozycji. Programistom doświadczonym wystarczy dokumentacja firmowa, opisująca dane wersje Logo. Fakt, że jest ona napisana w językach obcych, nie jest w tym przypadku istotną przeszkodą. Ale właśnie ci, dla których Logo będzie pierwszym językiem programowania, potrzebują tekstów mniej technicznych i bardziej zwracających uwagę na istotę budowy i użytkowania języka do rozwiązywania konkretnych problemów. Jest bowiem bardzo ważne, by właśnie pierwsze kontakty z komputerem były okazją do poznania właściwych metod posługiwania się nim i nabrania doświadczenia w prawidłowym metodycznie i skutecznym rozwiązywaniu różnych zagadnień.

W nauczaniu początkowym nie chodzi więc o coś w rodzaju kursu języka programowania i pamięciowe opanowanie jego konstrukcji, ale o coś znacznie ważniejszego – właśnie umiejętność skutecznego posługiwania się dobrze dobranymi metodami. Język programowania jest tu narzędziem i jako takie powinien być traktowany. Podobnie komputer. Nie jest na przykład celowe zajmowanie się w pierwszej fazie drobiazgowymi szczegółami budowy i działania komputera (chyba że chodziłoby o zaspokojenie zwykłej ciekawości), skoro można go użyć do rozwiązania zadania matematycznego bez wchodzenia w detale jego budowy i funkcjonowania.

W rozważaniach tego typu nasuwa się naturalne pytanie, czy można popełnić błąd metodyczny, polegający na nauczeniu niewłaściwych metod używania komputera, nieskutecznych, nadmiernie pracochłonnych w stosunku do uzyskiwanych efektów albo wręcz nie dających możliwości rozwiązywania zadań o większym stopniu komplikacji. Tak, i to znacznie częściej niżby się mogło początkowo zdawać. Najbardziej

szkodliwe są tu metody najbardziej przestarzałe, związane z najbardziej tradycyjnym podejściem do programowania, gdy prymitywizm komputerów ograniczał poważnie swobodę doboru naprawdę dobrych algorytmów postępowania. Najprymitywniejsze języki programowania, takie jak Basic, Fortran, Cobol czy też różne asemblery są właśnie przykładami tego, od czego nie należy zaczynać nauki w fazie początkowej. Słabe przystosowanie do wymagań narzucanych przez współczesne metody programowania, podatność konstrukcji w tych językach na elementarne błędy, zbyteczne absorbowanie programującego drobiazgowymi szczegółami technicznymi, ze szczegółami konstrukcji komputera włącznie, narzucanie rozmaitych zbędnych ograniczeń, kłopotliwych a nie zabezpieczających przed pomyłkami, gubienie struktur algorytmów i danych w gąszczu szczegółów – to wszystko stanowi o złej przydatności tych języków dla początkujących. Nadanie programom w tych niestrukturalnych językach właściwych struktur, tak żeby były przejrzyste, łatwe do sprawdzenia i modyfikacji, aby zapisany w nich algorytm był czytelny i zrozumiały, wymaga wiedzy i doświadczenia, które przekraczają możliwości początkujących. W rezultacie niewłaściwy wybór pierwszego języka i metod nauczania prowadzi nieuchronnie do wyrobienia od samego początku złych nawyków programowania i złego podejścia do rozwiązywania zadań na komputerze, gdyż w tej sytuacji o to najłatwiej i trudno osiągnąć coś więcej, gdy prymitywizm języka programowania i stosowanych metod nadmiernie absorbuje różnymi zbędnymi szczegółami technicznymi. Raz przyswojone nawyki trudno później wykorzenić, zresztą w większości przypadków nie ma już na to okazji. Tylko niewielki procent tych, którzy tak lub inaczej stykają się teraz z komputerami, ma okazję, chęć i możliwości opanowania później sztuki programowania na poziomie zawodowym, a w szczególności naprostowania raz nabytych złych przyzwyczajeń. Wszyscy pozostali mogą pozostać na raz osiągniętym prymitywnym poziomie wiedzy, ze swymi niewłaściwymi poglądami i nawykami, z niewątpliwą szkodą dla swoich możliwości prawidłowego korzystania z komputerów w swej pracy zawodowej lub w domu. Oto dlaczego złe uczenie informatyki jest gorsze od nieuczenia w ogóle. Oto dlaczego kwestia doboru pierwszego języka programowania ma tak istotną rolę.

Problem, jaki powinien być język programowania, który byłby najlepszy jako język pierwszego kontaktu z komputerem, został podjęty najwcześniej, bo jeszcze w latach sześćdziesiątych, w USA. Tak powstał język Logo. W pierwszej fazie prac brały udział tak znane ośrodki, jak Massachussetts Institute of Technology oraz Bolt, Beranek and Newman i tacy znani uczeni, jak: D. Bobrow, W. Feurzeig, C. Solomon i S. Papert. Po dziesięciu latach doświadczeń w szkołach i prac nad doskonaleniem języka oraz metodyką nauczania, prowadzonych przez grupę Seymoura Paperta w Artificial Intelligence Laboratory w MIT, powstała obecna wersja Logo, przejęta następnie i rozpowszechniona w niewiele różniących się wariantach przez różne firmy, zajmujące się produkcją i sprzedażą oprogramowania. W ten sposób Logo stało się dostępne jako szkolny język programowania na wszystkich bardziej rozpowszechnionych typach komputerów. W wielu krajach stworzono wersje Logo we własnych językach narodowych.

Mówiąc o handlowych wersjach Logo, trzeba ostrzec przed falsyfikatami. Nazwa nie jest strzeżona i wobec tego bywa przypisywana różnym niewydarzonym produktom, nie mającym nic wspólnego z właściwym językiem. Na szczęście, wobec tego że oryginalne Logo jest dość powszechnie dostępne, falsyfikaty istnieją dość krótko, odrzucane przez użytkowników, poznających się na ich kiepskiej wartości.

Określenie Logo jako języka początkowego nauczania programowania, przeznaczonego dla dzieci i młodzieży szkolnej, nie powinno stwarzać wrażenia, że nie jest to nic więcej, jak tylko prosta zabawka dla dzieci. W rzeczywistości jest to uniwersalny język programowania, w pełni konwersacyjny, o bardzo dużej sile ekspresji, wyposażony w wiele cech, których nie mają inne języki programowania. Jego bardzo dobrze przemyślana struktura czyni go dość łatwym i wygodnym w użyciu. Elementarne podręczniki Logo eksponują najbardziej jedną jego cechą, najbardziej widoczną w

początkach nauki języka, mianowicie „żółwią grafikę" (turtle graphics). Jest to dobrze przemyślana i bardzo wygodna w użyciu, zwłaszcza w trybie konwersacyjnym, wersja grafiki komputerowej, wymagająca tylko najprostszych środków technicznych – ekranu graficznego i klawiatury. W połączeniu z możliwością swobodnego definiowania procedur w dowolnym momencie stwarza to rzeczywiście bardzo sprawne narzędzie operowania obrazami na ekranie graficznym. Grafika żółwia została zresztą przejęta później przez wiele języków programowania, łącznie z Pascalem i Basicem, chociaż jest tam już ograniczona (z reguły) wskutek braku w tych językach możliwości tak spontanicznego definiowania procedur, jak to jest w Logo.

Ważną zaletą Logo jest również możliwość swobodnego wykonywania operacji na tekstach i strukturach listowych, dzięki czemu można, dobierając odpowiednio struktury danych dla konkretnych przypadków, tworzyć bardzo sprawne i zwarte algorytmy, które daje się ująć w formie krótkich i przejrzyście skonstruowanych procedur. Ponieważ zaś z prostych procedur można swobodnie budować znacznie bardziej skomplikowane, nawet przy niewielkiej wprawie można próbować programowania rozwiązań nawet bardziej złożonych problemów. Takiemu szybkiemu przechodzeniu od form prostych do złożonych sprzyja nie tylko pełna swoboda definiowania procedur w dowolnym momencie pracy, ale także dobry zestaw instrukcji języka oraz bardzo dobra ewidencja procedur, danych, plików, dająca w każdym momencie pełną kontrolę nad tym, co programista robi, bez obciążania go nadmiarem informacji zbędnych, które mu w danym momencie są niepotrzebne i nie muszą zaprzątać jego uwagi.

To wszystko i fakt, że Logo zostało już dokładnie przetestowane w szkołach, zanim jeszcze zyskało taką popularność, czyni go szczególnie interesującym również dla naszych szkół. Warto więc poznać go bliżej.

Podając opis języka i metod programowania w nim, musimy założyć, że Czytelnik ma dostęp do mikrokomputera z Logo i może się nim posługiwać w trakcie czytania tego tekstu. Jest to konieczne, gdyż bez komputera wiele uwag szczegółowych i objaśnień byłoby pozbawione jakiegokolwiek praktycznego sensu, sprowadzając się w najlepszym razie do abstrakcyjnych reguł o niejasnym znaczeniu. Zdając sobie sprawę, że wersje Logo na różnych typach mikrokomputerów różnią się nieco od siebie, musimy zdecydować się w tym opisie na wybór jednej z nich. Ze względu na to, że w szkołach najczęściej spotykanym obecnie typem jest Spectrum, wybieramy tę wersję języka. Ci, którzy mają inny typ mikrokomputera, mogą również posłużyć się tym opisem Logo, pomijając jedynie uwagi i informacje ściśle związane ze Spectrum. Każda omawiana niżej komenda Logo jest zawsze podawana w wersji pełnej oraz skróconej (jeśli jest). Formy skrócone komend najczęściej używanych w Logo są we wszystkich wersjach języka jednakowe i od nich należy rozpoczynać próby w przypadku wątpliwości. Pewne komendy mogą nie występować na innych mikrokomputerach lub mieć tam inną formę – wtedy najlepiej przykłady z nimi opuścić przy pierwszym czytaniu. Różnice między najczęściej występującymi w Polsce wersjami Logo zostały bardziej szczegółowo omówione w trzecim z cyklu moich artykułów o Logo w czasopiśmie „Informatyka" (nr 9 z 1985 r.).

W trakcie pisania tej książki powstała polska wersja Logo na Spectrum. Nie jest ona jednak jeszcze tak rozpowszechniona, by można było zrezygnować z wersji angielskiej, którą ma już bardzo wielu użytkowników Spectrum. Dlatego w tekście wszystkie przykłady są napisane w angielskim Logo SOLI/LCSI, natomiast polskie wersje wszystkich komend i komunikatów są podane na końcu książki.

Spectrum dla początkujących

Zadaniem tego rozdziału jest pomóc tym, którzy z Sinclair Spectrum stykają się po raz pierwszy. Ci, którzy już umieją nagrywać programy z magnetofonu i znają zasady posługiwania się klawiaturą mogą ten tekst opuścić.
Omówimy tu działanie i obsługę Spectrum w jego najmniejszym zestawie, który jest najbardziej popularny. Opis dotyczy obu wersji Spectrum: starszej, Spectrum 48k z gumową klawiaturą, i nowszej, Spectrum plus (rys. 2, 3).
Jak już było powiedziane we wstępie, Logo wymaga przynajmniej 48 kilobajtów pamięci o swobodnym dostępie, a więc wersja Spectrum 16k nie nadaje się. Jest możliwe rozszerzenie tej wersji do 48k, choć może to być operacja dość kosztowna.
Opakowanie firmowe mikrokomputera Sinclair Spectrum zawiera, w obu wersjach:

- mikrokomputer,
- zasilacz prądu stałego 220V/9V,
- kabel koncentryczny do łączenia gniazdka TV z gniazdem antenowym telewizora,
- parę złączonych kabelków magnetofonowych do połączenia gniazdek EAR i MIC z magnetofonem,
- książki: *Instrukcja obsługi* i *Podręcznik języka Basic,*
- kasetę z objaśnieniami i programami przykładowymi.

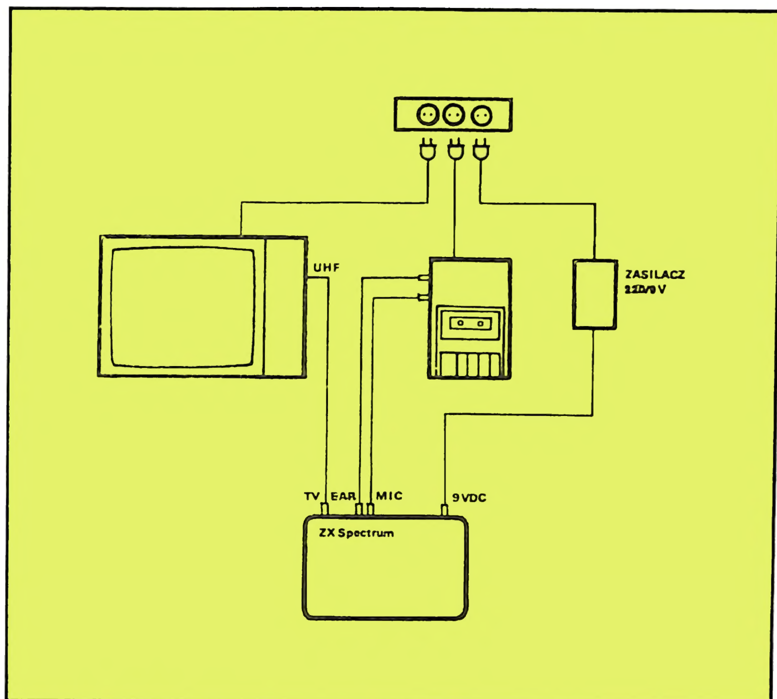
Oprócz tego potrzebny jest:

- telewizor z III zakresem OIRT (większość zwykłych telewizorów domowych ten warunek spełnia),
- magnetofon kasetowy.

W przypadku braku magnetofonu kasetowego można użyć innego. Magnetofon szpulowy też jest dobry, ale ma jeden istotny niedostatek; ponieważ z reguły oprogramowanie Spectrum jest dostępne na kasetach, trzeba by je uprzednio skopiować na taśmę. W każdym razie dobrze jest, jeśli magnetofon jest dobrej jakości, ale raczej nie stereofoniczny (jeśli mamy możność wyboru).
Zanim przystąpimy do łączenia wszystkiego według pokazanego schematu (rys. 1, 4 i 5), trzeba sprawdzić, czy wtyki kabli pasują do gniazdek telewizora (antenowe UHF) i magnetofonu. W każdym razie trzeba pamiętać o zasadzie:

> **WTYK ZASILANIA WŁĄCZA SIĘ DO KOMPUTERA JAKO OSTATNI, PO WYKONANIU I SPRAWDZENIU WSZYSTKICH INNYCH POŁĄCZEŃ.**
> (Jak to się robi, zaraz zobaczymy, na razie ta uwaga ma powstrzymać zbyt niecierpliwych).

Jeśli telewizor nie ma koncentrycznego gniazda antenowego, trzeba wykonać odpowiednie przejście. Najlepiej użyć właściwego łącznika z gniazdkiem koncentrycznym i wtyczką pasującą do telewizora, ale ostatecznie można to połączenie wykonać sposobem domowym. Trzeba przy tym pamiętać, że nie można dopuścić do zwarcia żył kabla, a połączenie musi być pewne i nie może się rozłączyć ani zewrzeć wskutek poruszania kabla. Szczegóły pozostawiamy pomysłowości Czytelnika.
Magnetofon powinien mieć gniazdka na dwuczęściowe wtyczki 3,6 mm. Jeśli ich nie ma, również trzeba wykonać odpowiednie przejście z elementów zakupionych w sklepie – wykonanie połączenia własnymi siłami jest raczej zbyt trudne.
Czasami magnetofon ma tylko jedno gniaz-



Rys. 1. Łączenie elementów układu mikrokomputera Spectrum

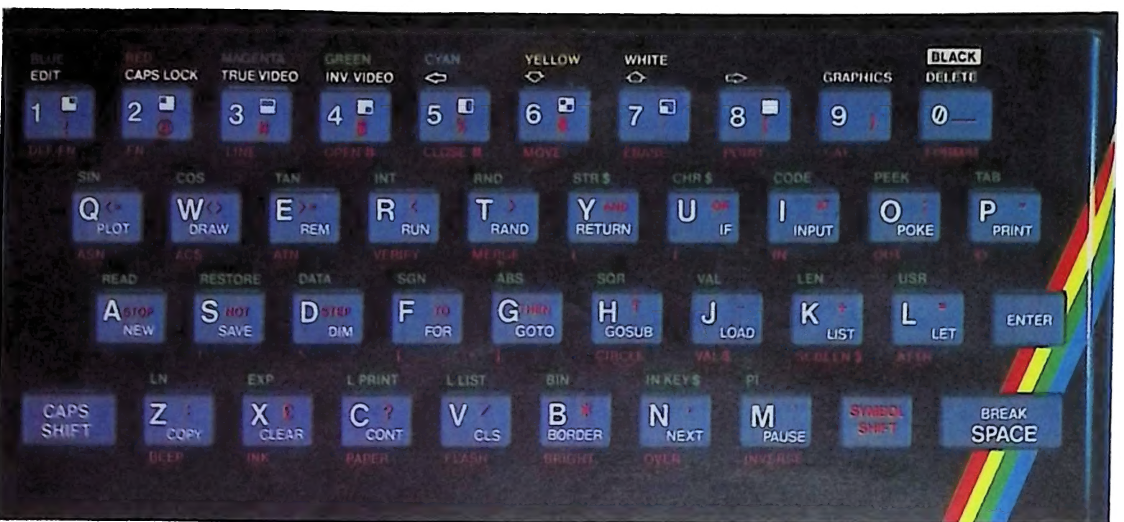
dko 3,6 mm – tak jest na przykład w popularnych magnetofonach produkcji Unitry. Do przegrywania programów z kasety do komputera to wystarczy – trzeba wetknąć dowolną z wtyczek kabelka magnetofonowego do tego gniazdka, a drugą wtyczkę tego samego koloru do gniazdka EAR na tylnej ściance komputera. Nie daje to jednak możliwości nagrywania z komputera na kasetę, ale na samym początku pracy nie będzie nam ono potrzebne.

Komputer powinien stać na równej i twardej powierzchni, tak aby był możliwy swobodny dostęp powietrza do otworków wentylacyjnych w jego obudowie. Nie należy go stawiać na suknie, kocu, obrusie itp., gdyż uniemożliwienie przepływu powietrza chłodzącego może doprowadzić przy dłuższej pracy do przegrzania komputera.

Spectrum plus ma składane nóżki, które umożliwiają ustawienie go w pozycji dogodniejszej do pisania na klawiaturze.

Kolorowy obraz można uzyskać na kolorowym telewizorze systemu PAL, a także telewizorze systemu SECAM ze specjalną przystawką. Bez przystawki telewizor systemu SECAM może być użyty jako czarno-biały.

Zasilanie telewizora, magnetofonu i komputera poprzez jego własny zasilacz wymaga trzech gniazdek sieciowych lub rozgałęźnika. Zapewnienie dostatecznej liczby gniazd sieciowych lub rozgałęźników jest ważne zwłaszcza wtedy, gdy ma być



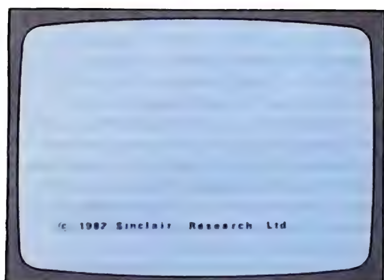
Rys. 2. Klawiatura Spectrum 48k

ale wystarczy nacisnąć dwa razy klawisz" (dolny rząd, 3 od lewej) – na ekranie powinno teraz być:

LOAD " "

- nacisnąć klawisz ENTER po prawej stronie klawiatury;
- włączyć magnetofon, naciskając klawisz odtwarzania.

Po starcie taśmy obrzeże ekranu zaczyna zmieniać barwy. Na telewizorze kolorowym będzie to na przemian kolor czerwony i niebieski, na czarno-białym jasny i ciemny. Po krótkim czasie na obrzeżu powinny pojawić



Rys. 6. Ekran po włączeniu komputera

się spokojnie przesuwające się regularne pasy czerwone i niebieskie, a w górze ekranu napis (rys. 7):

Program: LOGO

Po chwili pasy czerwono-niebieskie zostaną zastąpione nieregularnymi i szybciej przesuwającymi się paskami niebiesko-żółtymi (rys. 8). Jeśli z głośnika komputera słychać dźwięk, to w tym momencie staje się on nieregularny, o rytmie zgodnym z rytmem przesuwających się pasków. Może to trwać dość długo, po czym może znowu nastąpić ta sama sekwencja: regularne pasy niebiesko-czerwone i dźwięk jednostajny, a po nich nieregularne pasy niebiesko-żółte i niespokojny dźwięk. Na początku takiego okresu może się znowu ukazać napis na ekranie, dla danej taśmy Logo zawsze taki sam. Liczba takich sekwencji następujących po sobie pasów



Rys. 7. Czytanie z kasety nagłówka bloku Logo

niebiesko-czerwonych i niebiesko-żółtych zależy od wariantu taśmy Logo jakim dysponujemy; z reguły nie przekracza czterech. Czas ładowania: ok. 3 do 4 minut. Po prawidłowym załadowaniu Logo na ekranie znikają ruchome paski i pojawia się napis:

WELCOME TO SINCLAIR LOGO
© SOLI/LCSI 1984 VER 1.6

lub podobny, sygnalizujący gotowość Logo do pracy.
Wyłączamy magnetofon klawiszem STOP. Na razie nie będzie nam potrzebny.
Jeśli na ekranie ukaże się napis:

Tape loading error

(błąd ładowania taśmy), trzeba nagrywanie zacząć od początku.
Przyczyną błędu może być złe ustawienie



Rys. 8. Czytanie z kasety treści bloku Logo

regulatorów barwy bądź siły głosu. Dość często przyczyną zniekształceń formy sygnału jest przesunięcie regulatora barwy za bardzo w kierunku maksimum. Rzadziej – dobranie zbyt dużej siły głosu. Jeśli siła głosu była dobrana prawidłowo, paski na obrzeżu ekranu są wyraźne i nie zanikają – w ten sposób można kontrolować, czy regulator siły głosu nie został za bardzo przesunięty w kierunku minimum. Przyczyną słabego sygnału może być niedokładne wetknięcie wtyczek kabelka magnetofonowego do któregoś z gniazdek.
Zanim podejmiemy kolejną próbę nagrania, możemy wyzerować pamięć komputera, ścierając z niej to, co tam zostało zapisane. W tym celu trzeba nacisnąć klawisz A i potem ENTER. Na ekranie ukaże się napis:

NEW

a po naciśnięciu ENTER środkowa część ekranu ściemnieje i po chwili ukaże się ten sam napis, co bezpośrednio po włączeniu komputera. Na Spectrum plus ten sam efekt można uzyskać przez lekkie naciśnięcie przełącznika pod spodem z lewej strony obudowy. Dalej piszemy LOAD " " i wykonujemy wszystkie czynności jak poprzednio.
Jeśli nagranie ponownie się nie uda, trzeba uzbroić się w cierpliwość. Prawidłowe ustawienie regulatorów może wymagać kilku prób. Niektóre magnetofony mogą sprzęgać się z telewizorem lub komputerem, co może być jedną z przyczyn powstawania błędów nagrywania. Zmiana położenia magnetofonu względem innych elementów układu może pomóc. Dużą rolę odgrywa jakość magnetofonu. Możemy spróbować nagrać kasetę firmową. Jeśli to się uda, można spróbować jeszcze raz z kasetą Logo, pamiętając jednak, że poziomy nagrania obu kaset mogą być różne, a więc położenia regulatorów nie muszą być dokładnie takie same – mogą być lekkie odchylenia w jedną lub drugą stronę.
A co zrobić, gdy jeszcze przedtem nie udało się prawidłowo napisać LOAD " " albo NEW? Prawdopodobnie wtedy na ekranie są jakieś inne znaki i można je skasować. Na Spectrum plus służy do tego klawisz DELETE (po lewej stronie). Na gumowej klawiaturze osobnego takiego klawisza nie ma i trzeba nacisnąć i przytrzymać CAPS SHIFT (dolny rząd, lewa strona) oraz nacisnąć klawisz Ø (górny rząd, prawa strona). Przytrzymanie klawisza Ø spowoduje skasowanie wszystkich znaków aż do początku wiersza. Wtedy możemy napisać LOAD " ", po czym nacisnąć ENTER.
Na Spectrum plus można także ratować się naciśnięciem przełącznika zerującego po lewej stronie. Ale to jest ostateczność. Normalnie użycie DELETE zupełnie wystarczy.

Zasady bezpieczeństwa

Jako urządzenie zasilane elektrycznością, komputer jest równie bezpieczny jak radio bateryjne. Przy nieostrożnym obchodzeniu się człowieka z komputerem poszkodowany może być tylko ten ostatni. Celem podanych niżej uwag jest tylko to, by go nie zepsuć. Jeśli chodzi o pracujących przy komputerze, zwłaszcza młodzież, to należy tylko przestrzegać zwykłych zasad higieny pracy umysłowej i oglądania telewizji. Jasność i kontrast obrazu na telewizorze powinny być ustawione tak, aby nie męczyć wzroku obrazem zbyt jaskrawym albo zbyt ciemnym. Ekran powinien być ustawiony w takiej odległości, by można było wygodnie czytać napisy na nim - lecz nie bliżej. Nie należy siedzieć przy nim bez ruchu godzinami - przerwy w pracy komputerowi nie szkodzą. Pomieszczenie, w którym się pracuje, powinno być prawidłowo wietrzone. Komputer jest dobrze sprawdzony i przygotowany do pracy w normalnych warunkach i nie należy się obawiać, że każdy błąd w postępowaniu z nim musi spowodować jakieś uszkodzenie, jeśli tylko obchodzić się z nim będziemy ostrożnie i uważnie. Na pewno nie można uszkodzić komputera przez wpisywanie czegokolwiek z klawiatury ani dając mu do wykonania jakiekolwiek programy. Jeśli w czasie normalnej pracy zareaguje w sposób nieoczekiwany, trzeba spokojnie zastanowić się nad przyczyną takiej reakcji. Jeśli już absolutnie nie możemy sobie poradzić, wystarczy zacząć pracę od nowa. Niektóre klawisze, na przykład BREAK, właśnie służą do tego, by przerywać normalny tok pracy. Nieświadome dotknięcie takiego klawisza zamiast innego może spowodować skutki, w których w pierwszym momencie trudno się zorientować. W miarę nabywania doświadczenia takie przypadki przestają sprawiać kłopoty.
Uszkodzenia mechaniczne mogą być spowodowane nieuważnym lub brutalnym obchodzeniem się z komputerem: upadek ze stołu, rozbicie powłoki plastykowej, wyłamanie klawiszy, mechaniczne uszkodzenia gniazdek lub płaskiej łączówki. Inny typ uszkodzeń mechanicznych jest spowodowany zużyciem wskutek intensywnej eksploatacji: starcie napisów na i przy klawiszach, przetarcie lub zużycie połączeń elektrycznych klawiszy powodujące ich niepewne działanie lub niedziałanie przy naciskaniu, przesunięcie się klawiszy względem obudowy powodujące ich zacieranie się lub niepowracanie z położenia naciśniętego, rozregulowanie gniazdek powodujące niepewność połączeń z wtyczkami, przetarcia lub pęknięcia żył w kabelkach.
Uszkodzenia elektryczne mogą spowodować całkowity zanik prawidłowych reakcji komputera albo tylko ich ograniczenie. Na przykład może reagować z pozoru prawidłowo na włączenie i proste czynności, ale nie przyjmować albo nie wykonywać programów, które przedtem w takich samych okolicznościach nie sprawiały żadnych kłopotów, a mamy skądinąd pewność, że same programy uszkodzeniu nie uległy (na przykład w czasie nagrywania na magnetofon lub przegrywania z niego). Po stwierdzeniu, że wina na pewno leży po stronie komputera, a nie oprogramowania albo nieumiejętnej obsługi, trzeba zwrócić się do fachowca. Najczęstszymi przypadkami uszkodzeń tego typu jest przepalenie elementów elektronicznych albo uszkodzenie połączeń wewnątrz komputera. Dość wrażliwym miejscem jest gniazdo płaskiej łączówki z

tyłu obudowy. Niewłaściwe wkładanie doń łączówki, z przesunięciem względem właściwego położenia, może spowodować zwarcie kończące się przepaleniem elementów w komputerze. Elementy te mogą również zostać uszkodzone wskutek nieostrożnego dotykania pustego gniazda ręką i przeniesienia z niej do układów komputera elektrycznych ładunków statycznych. Może to się zdarzyć zwłaszcza w suchym powietrzu i gdy ma się na sobie ubranie z włóknami z tworzyw sztucznych. Zbyt duży ładunek statyczny może spowodować uszkodzenie delikatnych elementów elektronicznych. Można też uszkodzić komputer, przyłączając do gniazdka zasilania 9V niewłaściwe napięcie. Należy posługiwać się wyłącznie zasilaczem znajdującym się w kompletnym zestawie firmowym i tylko oryginalnymi wtyczkami, nie wchodzącymi do innych gniazd prócz zasilającego. Nie wolno posługiwać się kabelkami lub połączeniami powodującymi zwarcia lub przebicia. I wreszcie przyczyną uszkodzeń elektrycznych może być nieostrożne obchodzenie się z włączonym komputerem i nieprzestrzeganie zasad prawidłowego łączenia go z innymi urządzeniami i z siecią zasilającą.

Należy zwłaszcza przestrzegać zasady, że włączenie wtyku zasilania 9V do komputera jest możliwe tylko jako ostatnia czynność, po wykonaniu wszystkich innych połączeń i starannym ich sprawdzeniu, w tym także po włączeniu zasilacza do sieci 220V, zaś wyłączenie komputera przez wyjęcie tego wtyku zasilania 9V musi nastąpić przed rozłączeniem wszystkich innych połączeń, w tym także przed odłączeniem zasilacza od sieci 220V.

Od tej zasady może być jeden wyjątek: jeśli koniecznie musimy odłączyć magnetofon w czasie pracy komputera, to najpierw należy wyjąć wtyki kabelka łączącego z gniazdek EAR i MIC komputera, a potem ewentualnie odłączyć magnetofon od sieci – nigdy odwrotnie. Innych urządzeń poza magnetofonem nie należy przyłączać ani odłączać, jeśli komputer jest włączony. Przyłączając je poprzez płaską łączówkę należy upewnić się przed włączeniem komputera, że została prawidłowo włożona. Poza tym przy wtyku płaskim nie należy manipulować nigdy, a już zwłaszcza przy włączonym komputerze.

W razie zaniku lub wyraźnych wahań napięcia w sieci pierwszą czynnością powinno być wyjęcie wtyku zasilania 9V. Nie można dopuścić, by wtyk ten był w gniazdku zasilania komputera w momencie dołączania napięcia sieci.

Nie należy używać wylatujących ze ścian lub iskrzących gniazdek sieciowych.

Należy prowadzić wszystkie kable łączące tak, aby nie było możliwe ich przypadkowe zerwanie lub rozłączenie przy przechodzeniu, wstawaniu od stołu lub normalnym poruszaniu krzeseł czy innych mebli, zwłaszcza przez osoby nie uprzedzone, małe dzieci lub zwierzę domowe.

Po rozłączeniu połączeń kabli nie należy załamywać ani owijać wokół zasilacza lub komputera. Przechowywanie ich w porządku uchroni nas także przed sytuacją, gdy nie można korzystać z komputera i staje się on bezużyteczny z powodu zagubienia jednego kabelka.

Przyłączanie urządzeń nietypowych, jak na przykład drukarek lub stacji dyskietek, które nie są dostarczane przez firmy handlujące komputerami jako sprzęt nadający się do użycia ze Spectrum, albo telewizorów kolorowych systemu innego niż PAL, wymaga uprzedniej konsultacji i nadzoru kompetentnego fachowca lub godnej zaufania firmy i w każdym przypadku należy się stosować ściśle do ich wskazówek.

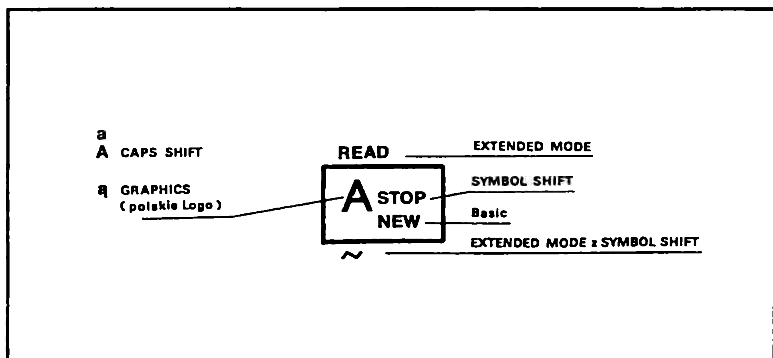
Warto jeszcze raz podkreślić, że przestrzeganie tych wszystkich wskazówek ma na celu stworzenie niezbędnego marginesu bezpieczeństwa i choć przypadkowe złamanie którejś z nich nie musi od razu powodować awarii komputera, to jednak stosowanie się do tych zasad powinno być elementem dobrej praktyki użytkownika Spectrum. Wiele uszkodzeń tych komputerów było spowodowanych elementarną nieostrożnością, polegającą właśnie na przekroczeniu tych prostych reguł, choć oczywiście także w wielu wypadkach komputery to jakoś wytrzymywały i do awarii nie dochodziło. Naprawy są jednak dość kosztowne i znacznie prościej jest nie stwarzać do nich powodu.

Logo na Spectrum: pierwsze kroki

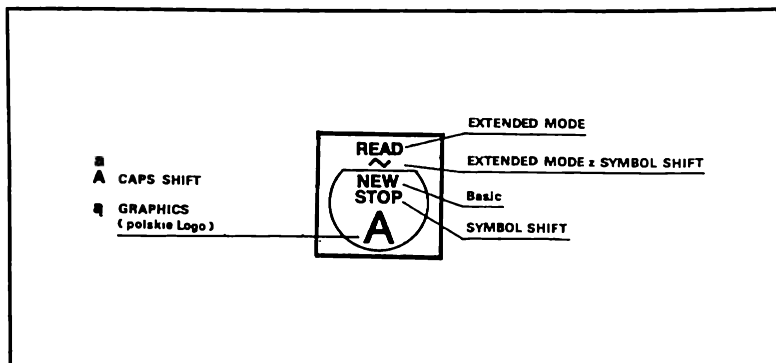
Po wczytaniu Logo na ekranie mamy napis:

WELCOME TO SINCLAIR LOGO
© SOLI/LCSI 1984 VER 1.6

i w następnym wierszu pytajnik, a zaraz po nim migający wskaźnik, oznaczający gotowość przyjęcia komendy Logo. W prawym dolnym rogu jest litera l, pokazująca, że z klawiatury będą przyjmowane małe litery. Jeśli mamy do czynienia z inną wersją Logo, na przykład polską, napis będzie inny, ale litery w prawym dolnym rogu ekranu, wskazujące stan klawiatury, będą dla wszystkich wersji takie same.
Tryb przyjmowania znaków z klawiatury można zmienić, naciskając klawisze CAPS SHIFT, SYMBOL SHIFT, które są w obu wersjach komputerów Spectrum. Rys. 9 i 10 pokazują dla wybranego klawisza odpowiedniość między oznaczeniami na klawiaturze i trybem wprowadzania znaków. Widać, że oprócz wymienionych są jeszcze tryby EXTENDED MODE oraz EXTENDED MODE z SYMBOL SHIFT, ale te na razie nie będą nam potrzebne. O ich użyciu dowiemy się znacznie później, przy opisie edytora Logo. Tryb GRAPHIC jest potrzebny w polskim Logo i służy do tego, by dodać literze znak diakrytyczny: ogonek dla a, e lub kreskę dla c, n, o, s, z, l. Litery ż i ź mają w polskim Logo taką samą postać. W wersji angielskiej takiej możliwości tworzenia polskich liter nie ma i tryb GRAPHICS nie będzie potrzebny, przynajmniej na początku.
Aby przejść do trybu GRAPHICS na gumowej klawiaturze, trzeba nacisnąć i przytrzymać CAPS SHIFT, po czym nacisnąć klawisz 9. Klawiatura przechodzi w stan



Rys. 9. Związek oznaczeń klawisza z trybami wprowadzania znaków w Spectrum 48k



Rys. 10. Związek oznaczeń na klawiszu z trybami wprowadzanie znaków w Spectrum +

GRAPHICS na czas pisania następnego znaku, po czym wraca do stanu poprzedniego. Przejście do stanu GRAPHICS sygnalizuje litera G w prawym dolnym rogu. Podobnie jest z przejściem do stanu EXTENDED MODE, tylko trzeba nacisnąć CAPS SHIFT (i przytrzymać) a potem SYMBOL SHIFT, a przejście do tego stanu sygnalizuje litera E w prawym dolnym rogu. Na Spectrum plus używamy do tego klawiszy GRAPH i EXTENDED MODE..

Przejście do trybu pisania dużych liter na stałe zapewnia naciśnięcie CAPS SHIFT i potem CAPS LOCK (klawisz 2, górny rząd, lewa strona) na gumowej klawiaturze lub CAPS LOCK na Spectrum plus. Wtedy w prawym dolnym rogu ekranu pojawi się C, a z klawiatury będą przyjmowane duże litery. Ponowne naciśnięcie tych samych klawiszy przełączających spowoduje powrót do trybu małych liter z I w prawym dolnym rogu ekranu.

Naciśnięcie samego CAPS SHIFT albo samego SYMBOL SHIFT nie powoduje zmiany litery w prawym dolnym rogu, ale dopóki taki klawisz jest naciśnięty z klawiatury będą wprowadzane duże litery albo znaki napisane na klawiszu przy literze (na gumowej klawiaturze czerwone). Znaki złożone <= <> >= i słowa napisane na klawiszu nie są wyprowadzane. Nawiasy kwadratowe uzyskuje się naciskając SYMBOL SHIFT i jeden z klawiszy Y albo U – przejście do trybu EXTENDED MODE nie jest przy tym potrzebne.

Zaczynamy więc od trybu I. Jeśli Logo zostało prawidłowo wprowadzone i na ekranie jest pytajnik z migoczącym wskaźnikiem:

?

możemy napisać:

clearscreen

albo w skrócie:

cs

i nacisnąć klawisz ENTER. Na środku ekranu pojawia się żółw Logo: trójkątny znaczek, którego jeden wierzchołek, wyraźnie oznaczony, jest skierowany do góry (rys. 11). Pytajnik z migającym wskaźnikiem, sygnalizujący gotowość do przyjęcia komendy, pojawia się w dole ekranu. W prawym dolnym rogu ekranu jest nadal litera I, wskazująca stan klawiatury: pisanie małych liter.

Posuńmy żółwia o 40 kroków w przód, pisząc forward lub fd z podaną liczbą kroków:

fd 40 ENTER

Pomiędzy fd a liczbą kroków musi być

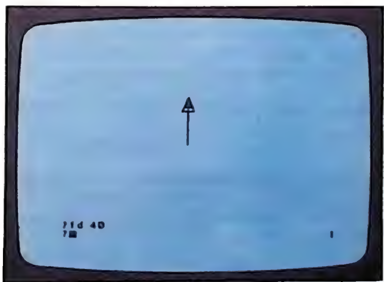


Rys. 11. Ekran po clearscreen

odstęp: dość częstym błędem początkujących jest zapominanie o tym i pisanie razem fd40, co jest niepoprawne. Pomyłka tego typu nie powoduje żadnych istotnych skutków. Po uwadze:

I don't know how to FD40

(czyli: nie wiem, jak wykonać FD40) naciskamy ENTER i znowu pojawia się pytajnik z migającym wskaźnikiem, po czym możemy napisać komendę poprawnie. Otrzymamy wynik jak na rys. 12.



Rys. 12. Przesunięcie żółwia naprzód o 40 kroków

Obróćmy teraz żółwia w prawo (right lub rt) o 60 stopni (pamiętamy o odstępie!):

rt 60 ENTER

(rys. 13) i przesuńmy w przód:

fd 60 ENTER



Rys. 13. Obrót żółwia w prawo o 60 stopni

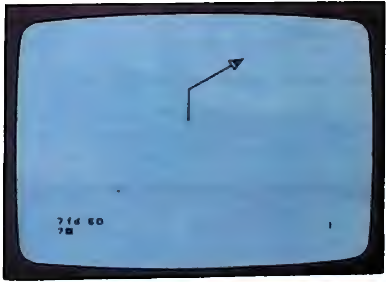
(rys. 14), a następnie w tył o 100 kroków (używając back lub bk):

bk 100 ENTER

(rys. 15), po czym obróćmy w lewo (left lub lt):

lt 90 ENTER

Na końcu tego żurawia umieśćmy przeciwwagę w postaci kwadratu o boku 10. Będzie to okazja do wprowadzenia procedury rysującej taki kwadrat. Wybierzmy dla niej nazwę i napiszmy ją poprzedzając słowem to:



Rys. 14. Naprzód 60

to kwadrat ENTER

Zauważmy, że po napisaniu tego na ekranie pojawił się inny wskaźnik: >



Rys. 15. Wstecz 100

oznaczający gotowość do przyjęcia kolejnego wiersza procedury, bez zmieniania położenia żółwia i rysunku na ekranie (rys. 16). Aby żółw narysował taki kwadrat, musi czterokrotnie pójść naprzód 10 kroków i zrobić zwrot o 90 stopni w prawo:

repeat 4 [fd 10 rt 90] ENTER

Jak pamiętamy, nawiasy kwadratowe piszemy naciskając SYMBOL SHIFT i Y (nawias otwierający) lub U (nawias zamykający). Teraz możemy zakończyć definiowanie tej procedury:

end ENTER

Na ekranie pojawia się napis:

KWADRAT defined

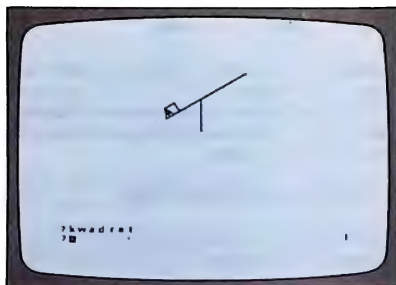


Rys. 16. Początek definicji procedury

oznaczający, że słowo kwadrat może być użyte jako komenda Logo. W następnym wierszu zamiast poprzedniego > pojawia się znów pytajnik z migoczącym wskaźnikiem, sygnalizujący gotowość przyjęcia komendy. Napiszmy więc:

kwadrat ENTER

i zobaczmy co wyszło (rys. 17).
Dotychczas żółw po każdym przesunięciu pozostawiał kreskę. Możemy powiedzieć, że wędrował z pisakiem, który był opuszczony i dlatego pozostawiał ślad. Możemy kazać podnieść ten pisak, używając komendy penup lub pu, oraz posłać żółwia do miejsca, z którego rozpoczął swą podróż:



Rys. 17. Wywołanie procedury

pu ENTER
home ENTER

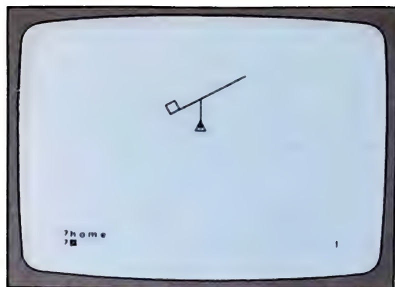
Po tej ostatniej komendzie żółw znów siedzi w środku ekranu, skierowany do góry (rys. 18). Opuśćmy mu pisak (pendown lub pd) i skierujmy w lewo:

pd ENTER
lt 90 ENTER

po czym każmy mu posunąć się do przodu o dużą liczbę kroków:

fd 400 ENTER

Widzimy, co się stało: żółw, wybiegając w lewo poza brzeg ekranu pojawia się natychmiast na prawym brzegu i biegnie dalej.



Rys. 18. Powrót żółwia do pozycji wyjściowej

Lewy i prawy oraz górny i dolny brzeg ekranu są sklejone parami, tak że ekran jest homeomorficzny z torusem.
Z miejsca, w którym żółw się znajduje, możemy go poprowadzić pod prawy koniec żurawia i założyć tam studnię (rys. 19):

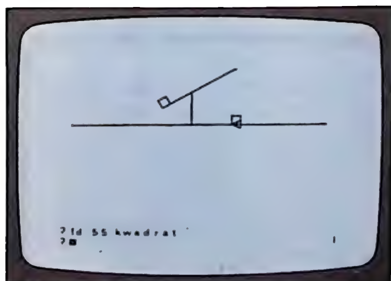
fd 55 kwadrat ENTER

Po obiegnięciu kwadratu żółw jest w jego prawym dolnym rogu, skierowany w lewo. Teraz można na ramieniu żurawia powiesić linkę – przesunąć żółwia na środek studni, obrócić w prawo:

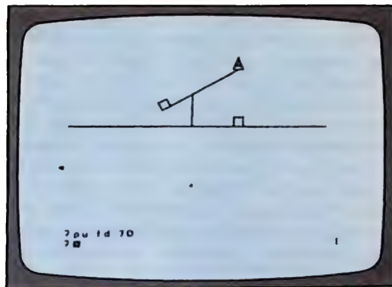
fd 5 rt 90 ENTER

po czym z podniesionym pisakiem ustawić go mniej więcej na końcu żurawia (rys. 20):

pu fd 70 ENTER



Rys. 19. Sytuacja po przesunięciu żółwia i wywołaniu procedury kwadrat



Rys. 20. Ruch bez rysowania: przesunięcie żółwia naprzód z podniesionym pisakiem

Te przykłady pokazują, że kolejne komendy można pisać w tym samym wierszu, oddzielając je od siebie odstępami. W takim przypadku naciśnięcie klawisza ENTER oznacza żądanie wykonania wszystkich komend tego wiersza po kolei. Przyjmując, że Czytelnik już się przyzwyczaił do tego, że po napisaniu każdego wiersza naciska się ENTER, nie będziemy więcej o tym przypominać – symbole ENTER na końcu wiersza będziemy opuszczać.
Środek podstawy trójkąta-żółwia powinien teraz być na końcu skośnego odcinka. W tej sytuacji żółw może zjechać w dół, rozwijając za sobą linkę na podobieństwo pająka:

rt 180 pd fd 55

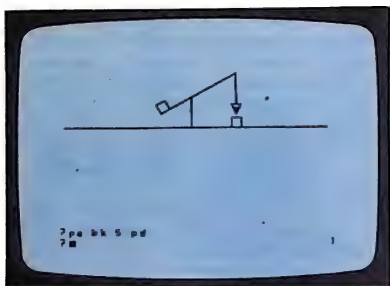
Jeśli uznamy, że linka jest za długa i część jej trzeba zmazać, użyjemy penerase lub pe, co oznacza przejście pisaka w stan ścierania kreski. Użycie potem pendown lub pd wznawia znów możliwość rysowania:

pe bk 5 pd

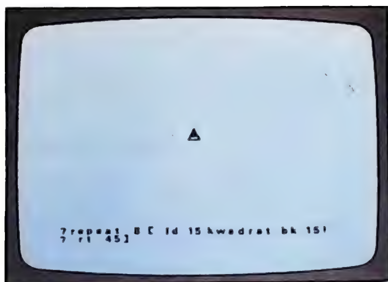
Linka stała się o 5 jednostek krótsza (rys 21). Teraz możemy spróbować narysować na jej końcu wiadro, ale wykonanie tego pozostawiamy pomysłowości Czytelnika.
Poznane przy okazji rysowania kwadratu powtarzanie można wykorzystać do rysowania różnych wzorków okresowych. Może to być na przykład pęk chorągiewek (rys. 22, 23):

repeat 8 [fd 15 kwadrat bk 15 rt 45]

Przekroczenie pojemności jednego wiersza ekranu powoduje automatyczne przejście do następnego wiersza. W tym przypadku następuje to po bk 15, czyli po 30 znakach. Bezpośrednio po nich pojawia się automatycznie na 31 miejscu wiersza wykrzyknik !, jako znak przejścia do następnego wiersza. Niezależnie od tego, że całość mieści się w dwóch wierszach ekranu, jest to jeden i ten sam wiersz Logo i klawisz ENTER naciska się dopiero po jego zakończeniu, to znaczy po zamknięciu nawiasu kwadratowego. Naciśnięcie ENTER wcześniej, przed napisaniem tego nawiasu powoduje automatyczne jego dopisanie. Przedwczesne ENTER byłoby żądaniem wykonania niekompletnej komendy i musiałoby się skończyć sygnalizacją błędu, albo też, jeżeli taka niedokończona komenda po automatycznym uzupełnieniu na końcu kwadratowym nawiasem zamykającym staje się poprawna w sensie Logo, to spowoduje ona wykonanie podaną liczbę razy tej sekwencji komend, która jest zawarta w nawiasach kwadratowych. Jak widać, Sinclair Logo nie jest zbyt odporne na błędy tego typu i dlatego warto zapamiętać regułę: komenda repeat musi się zmieścić cała w jednym wierszu Logo. Ponieważ na końcu wiersza Logo jest zawsze ENTER (i nigdy nie może być w środku), sformułowanie to oznacza zakaz naciskania tego klawisza przed wprowadzeniem ostatniego nawiasu zamykającego komendy repeat. Choć wiersz Logo może zawierać ponad 6 wierszy ekranu, warto dążyć do tego, by zapisy nie były za długie i zbyt skomplikowane sekwencje komend zastępować procedurami.

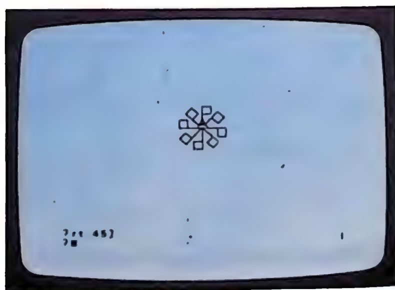


Rys. 21. Przykład ścierania części odcinka



Rys. 22. Złożona komenda Logo przed wykonaniem

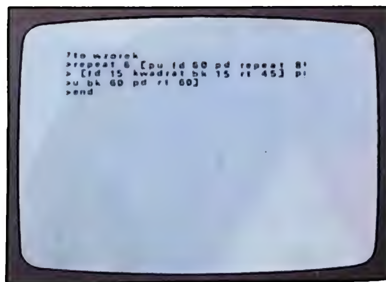
W trybie graficznym ekranu, gdy większa jego część jest przeznaczona na rysunek i na wprowadzany tekst są przeznaczone tylko dwa dolne wiersze, zbyt długie ciągi komend lub długie pojedyncze komendy Logo mogą się w tych dwu wierszach nie zmieścić: początek takiego długiego wiersza Logo może zniknąć, podczas gdy jego koniec nie został jeszcze napisany. Aby tego uniknąć, można przejść przedtem do trybu tekstowego, gdy cały ekran jest przeznaczony na tekst i mogą się na nim zmieścić 22 wiersze ekranowe po 30 znaków każdy. Uzyskujemy to pisząc komendę textscreen, lub w skrócie ts. Po jej wykonaniu pytajnik z migoczącym wskaźnikiem pojawia się na początku pierwszego wiersza, to znaczy w lewym górnym rogu ekranu. Wskaźnik stanu klawiatury jest, jak



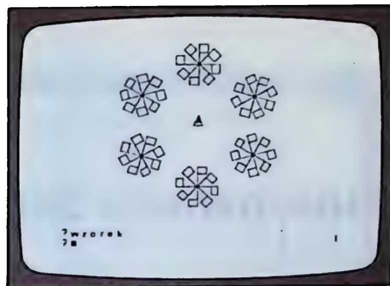
Rys. 23. To samo po wykonaniu: pęk chorągiewek

poprzednio, w prawym dolnym rogu. Poprzednia zawartość ekranu, zarówno tekst jak rysunek, zostaje skasowana.
Oznacza to, że nie można użyć textscreen, jeżeli ma się na ekranie rysunek, którego się nie chce zniszczyć. Pod tym względem Sinclair Logo jest gorsze od innych wersji. Na przykład Apple Logo, na którym jest ono wzorowane, ma jakby dwie strony ekranowe, tekstową i rysunkową, i można je dowolnie przełączać tak, że pojawienie się jednej nie niszczy drugiej. Czytelnik, mający dostęp do jakiejkolwiek innej wersji Logo, może zbadać to przełączanie stron eksperymentalnie.
Najlepszym sposobem zachowania rysunku, a dokładniej uczynienia go odtwarzalnym jest jednak zdefiniowanie procedury rysowania go. W takim przypadku możemy użyć textscreen bez wahania (rys. 24):

ts
to wzorek
repeat 6 [pu fd 60 pd repeat 8 !
[fd 15 kwadrat bk 15 rt 45] pu !
bk 60 pd rt 60]
end



Rys. 24. Definiowanie procedury po ts. Przykład długiego wiersza Logo



Rys. 25. Wynik wywołania procedury wzorek

Zgodnie z podaną wyżej regułą pisania komend repeat procedura wzorek składa się z 3 wierszy Logo (nagłówka: to wzorek jednego wiersza treści i końcowego wiersza: end), ale na ekranie zmieści się w pięciu wierszach. W naszym tekście potrzeba na to 6 wierszy, bo łam jest za wąski, ale tak samo jak na ekranie, wykrzyknik na końcu wiersza oznacza przejście do następnego.
Przejście od trybu tekstowego do trybu rysowania na ekranie następuje automatycznie w momencie użycia jakiejkolwiek komendy powodującej przesuwanie żółwia lub w ogóle mającej coś do czynienia z jego lokalizacją, doborem koloru rysowanej przezeń kreski, doborem koloru tła rysunku lub obrzeża ekranu. Pisząc:

wzorek

otrzymujemy obraz jak na rys. 25. Oczywiście procedura jest prawidłowo zapamiętana i można z niej dowolnie korzystać mimo tego, że znikła z ekranu w momencie wykonywania tej komendy, powodującej przejście do rysowania pęków chorągiewek.

Operowanie barwami

Dotychczas wszystkie rysunki były rysowane czarną linią na białym tle. Również teksty były pisane czarno na białym. Można jednak dobierać niezależnie kolory każdego z następujących elementów obrazu na ekranie:

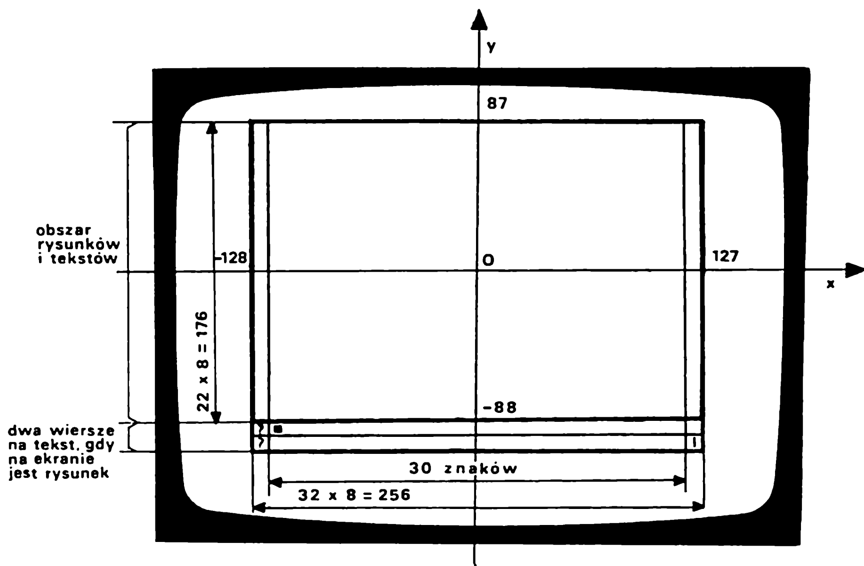
- tła rysunku,
- linii rysunku lub punktu,
- tła tekstu,
- litery tekstu,
- obrzeża.

Mając kolorowy ekran dysponujemy ośmioma różnymi kolorami. Na ekranie czarno-białym (monochromatycznym) będą one widoczne jako różne odcienie szarości. Są one oznaczone numerami od 0 do 7 – nad klawiszami z tymi cyframi są napisane angielskie nazwy odpowiednich kolorów (rys. 26):

0 – czarny – black
1 – niebieski – blue
2 – czerwony – red
3 – purpurowy – magenta



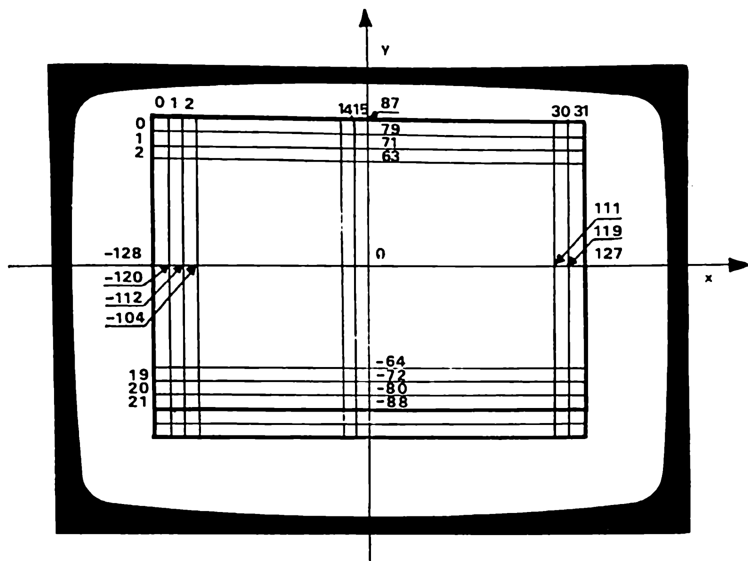
Rys. 26. Kolory ekranu (Uwaga: rzeczywiste kolory mogą zależeć od typu telewizora, jego regulacji, stopnia zakłóceń itp.)



Rys. 27. Układ współrzędnych i rozmiary ekranu

4 – zielony – green
5 – błękitny – cyan (light blue)
6 – żółty – yellow
7 – biały – white

Na ekranie monochromatycznym odcień jest tym jaśniejszy, im wyższy jest jego numer.
Do nadawania kolorów służą następujące komendy:
setbg – kolor tła o numerze podanym jako argument komendy, na przykład setbg 4 da tło koloru zielonego;
setpc – kolor kreski rysowanej przez żółwia, kropek wchodzących w skład rysunku i kolor samego wizerunku żółwia;
settc – kolor tła i liter tekstu; argumentem komendy jest para numerów kolorów (w tej kolejności), ujęta w nawiasy kwadratowe;
setbr – kolor obrzeża – komendy tej można również użyć w postaci nie skróconej: setborder.
Dla wszystkich tych komend prócz ostatniej, setbr, stosuje się tylko podane tu formy skrótowe. Użycie form pełnych: setbackground, setpencolour, settextcolour byłoby w Spectrum Logo niepoprawne.
Podanie komendy setbg lub setpc w trybie tekstowym powoduje przejście do trybu rysowania z żółwiem w pozycji home, to znaczy w środku ekranu, skierowanym w górę. Jednocześnie zostaje ustawiony właściwy kolor pisaka i wizerunku żółwia albo tła. Te same komendy podane w trybie rysowania powodują zmianę odpowiedniego koloru, ale rysunek już istniejący na ekranie może być przy tym zniekształcony. Zmiana koloru tła może spowodować zanik niektórych linii i punktów rysunku, natomiast zmiany koloru linii i koloru wizerunku żółwia w trakcie rysowania mogą spowodować powstanie efektu rozlewania się kolorów. Jest to związane z przyjętą zasadą obsługi kolorowego ekranu przez mikrokomputer Spectrum.
Cały ekran jest podzielony na trzy części: obrzeże, przestrzeń na rysunek lub tekst (w trybie tekstowym) oraz bezpośrednio pod



Rys. 28. Numeracja kolumn i wierszy tekstu na tle układu współrzędnych

nią miejsce na dwa wiersze tekstu, wykorzystywane wyłącznie w trybie rysowania (rys. 27). Miejsce na rysunek ma wymiary 256 na 176 jednostek, czyli zakresy współrzędnych: x od -128 do 127, y od -88 do 87 (rys. 28). Każdy znak mieści się w kwadracie 8 na 8 jednostek, a więc na tej samej przestrzeni mogą zmieścić się 22 wiersze po 32 znaki każdy. W trybie tekstowym można jednak wprowadzić z klawiatury do każdego wiersza tylko 30 znaków, gdyż jedno miejsce na początku wiersza jest zarezerwowane na stałe na ukazujący się tam pytajnik - znak gotowości, a ostatnie miejsce wiersza jest przeznaczone na wykrzyknik !, jako znak kontynuacji, jeżeli wiersz Logo będzie dłuższy niż wiersz ekranu. Jeśli tekst jest drukowany przez komputer, wszystkie 32 miejsca wiersza sa jednakowo dostępne. Tak samo jest przy tworzeniu rysunków.

W każdym takim kwadracie 8 na 8 wszystkie kreski i punkty mogą mieć tylko jeden kolor i jest to kolor elementu, który się tam pojawił jako ostatni: fragment rysunku albo nawet obraz żółwia, jeżeli tylko ten miał okazję zatrzymać się między wykonaniem dwu komend tak, że przynajmniej fragment jego wizerunku wypadł w tym kwadracie. Kolor tła jest we wszystkich kwadratach taki sam.

Jak z tego wynika, efekt rozlewania barw jest zupełnie niezauważalny w przypadku rysowania rysunku liniami jednego koloru albo nawet wtedy, gdy części rysunku o różnych kolorach są dostatecznie odległe od siebie tak, aby nie znaleźć się w jednym i tym samym kwadracie 8 na 8, a żółw będzie się w trakcie rysowania poruszać tak, aby nie rozlewać farby tam, gdzie nie trzeba.

Efekt rozlewania barw jest najbardziej widoczny wtedy, gdy na rysunku są obszary całkowicie zamalowane jednym kolorem. Można to uzyskać przez pokrycie takiego obszaru kreskami kładzionymi równolegle w odległości 1. Nie jest przy tym istotne, czy kolor kreski jest różny od koloru tła. W takiej sytuacji wtrącenie obcego koloru

nawet w jednym punkcie może powodować zmianę barwy całych kwadratów.
To wszystko, co zostało tu powiedziane, można stosunkowo łatwo sprawdzić, zamalowując pasy o szerokości ośmiu jednostek i umieszczając obok siebie takie pasy o różnych kolorach. Zróbmy to tak, aby komplet takich pasów zarówno w położeniu poziomym, jak pionowym mieścił się swobodnie w polu przeznaczonym na rysunek, gdy jego środek wypada w początku układu współrzędnych. Niech więc taki komplet ma wymiary 176 na 176, czyli składa się z 22 pasów o szerokości 8 i długości 176. Numer koloru każdego pasa będzie losowany. Skorzystamy w tym celu z komendy random 8, która generuje losowo liczbę naturalną z przedziału pomiędzy 0 i 7:

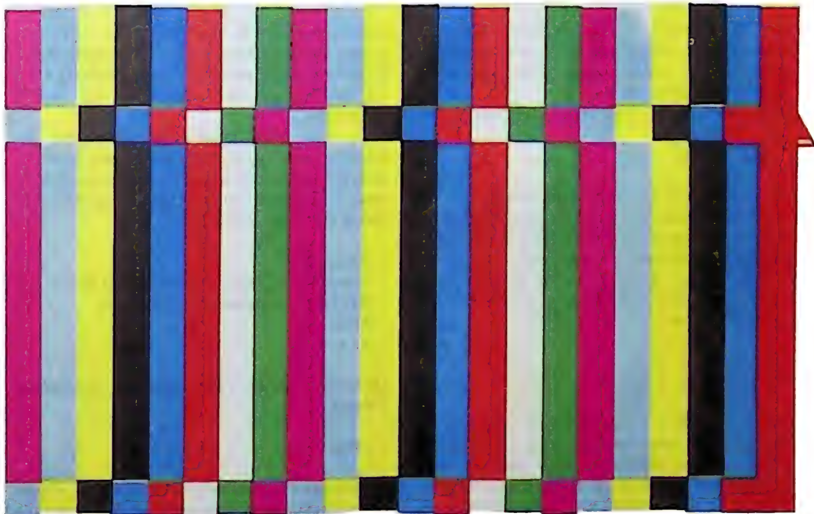
```
ts
to pas
setpc random 8
repeat 8 [fd 87 bk 175 fd 88 pu rt 90 fd !
1 lt 90 pd]
end
```

```
to pasy
pu lt 90 fd 88 rt 90 pd
repeat 22 [pas]
end
```

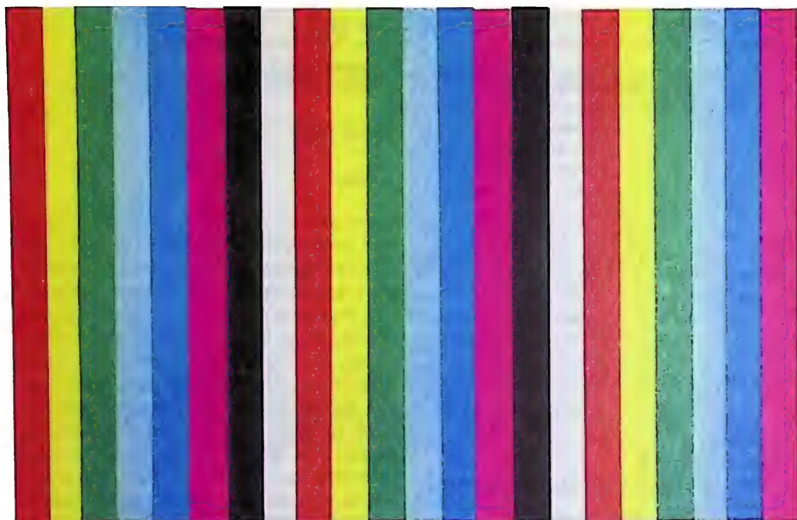
Pisząc teraz:

```
pasy
```

zobaczymy, że rzeczywiście powstaje komplet pionowych pasów, ale w środku i na dole każdego pojawia się kwadrat w kolorze pasa sąsiadującego z prawej strony (rys. 29). Powstają one wskutek tego, że żółw, rysując nowy pas, zatrzymuje się w jego środku i na krańcach, a w pierwszej fazie rysowania jeszcze wystaje poza rysowany pas na pas poprzedni. Efekt ten zniknie, jeżeli pozbędziemy się wizerunku żółwia z ekranu (rys. 30). Służy do tego komenda hideturtle lub w skrócie ht. Ponowne pojawienie się żółwia na ekranie może spowodować komenda showturtle lub w krócie st.



Rys. 29. Wynik wykonania procedury pasy z obrazem żółwia na ekranie (fragment). Widać efekt rozlewania barw.



Rys. 30. Ten sam obraz rysowany ukrytym żółwiem.

Skutki wszystkich pozostałych komend dotyczących żółwia są takie same niezależnie od tego, czy jego wizerunek jest widoczny na ekranie czy nie.
Wykonajmy więc sekwencję komend:

```
cs ht pasy
```

Po wykonaniu tego żółw jest na prawo od skrajnego prawego pasa, w jego środku, w punkcie o współrzędnych 88 i 0. Możemy go ukazać, obrócić w lewo, znowu schować i narysować drugi komplet pasów, tym razem poziomych. Zaobserwujmy efekt rozlewania barw:

```
st lt 90
ht pasy
```

To, że każdy z narysowanych pasów ma szerokość 8, wynika tylko stąd, że lewy (lub dolny) brzeg każdego pasa miał współrzędną podzielną przez 8, czyli wypadał na skraju rzędu kwadratów. Uzyskaliśmy to dzięki temu, że współrzędna x żółwia przed rysowaniem pasów pionowych i współrzędna y przed rysowaniem pasów poziomych były podzielne przez 8 (dokładniej, były równe 0). Czytelnik może sprawdzić, co się stanie, gdy ten warunek nie będzie spełniony.
Żeby nie ograniczać się wyłącznie do testów, spróbujmy napisać procedurę rysującą kolorowy zygzak. Załóżmy, że będzie on miał 10 ząbków, kolor każdego odcinka jest losowany (rys. 31, 32):

```
to zygzak
repeat 10 [setpc random 8 fd 8 rt !
60 setpc random 8 bk 8 lt 60]
end
st zygzak
```

Oczyśćmy ekran, nie zmieniając położenia żółwia:

```
clean
```

Ponieważ kolor żółwia był losowany, może się zdarzyć, że w tym momencie będzie on niewidoczny, mimo wykonania przedtem st, jeżeli przypadkiem kolor żółwia wypadł



Rys. 31. Przykład zygzaka o boku 8 na tle siatki 8 × 8. Tak by to wyglądało, gdyby nie było rozlewania barw.



Rys. 32. A tak to wygląda w rzeczywistości

taki sam, jak kolor tła. Narysujmy jeszcze kilka takich zygzaków :

zygzak
penup fd 40 pd

repeat 3 [rt 120 zygzak]

Spróbujmy dalej rysować te kolorowe zygzaki tak, aby się przecinały i zobaczmy, jaki będzie skutek. Możemy również eksperymentować ze zmianą koloru tła.

Rozmieszczanie i barwy tekstów

Przy pisaniu znaków efektu rozlewania barw nie ma, bo każdy znak jest zawsze w odrębnym kwadracie. Teksty mogą być pisane w sposób ciągły, tak jak to widzieliśmy cały czas. Można również wpisywać znaki lub sekwencje znaków w wybranych miejscach ekranu, zaczynając od kwadratu o zadanych współrzędnych. Współrzędne kwadratów w polu przeznaczonym na rysunki i teksty są następujące: numery kolumn od 0 do 31, numery wierszy od 0 do 21. Jak już było powiedziane wyżej, kolumna 31 jest w zasadzie przeznaczona na znak kontynuacji wiersza I, a kolumna 0 na znak gotowości do przyjęcia wiersza tekstu, czyli pytajnik. Tak jest przy wprowadzaniu komend. Natomiast komendy wyprowadzające na ekran teksty z pamięci mogą wykorzystywać wszystkie kolumny bez ograniczeń.

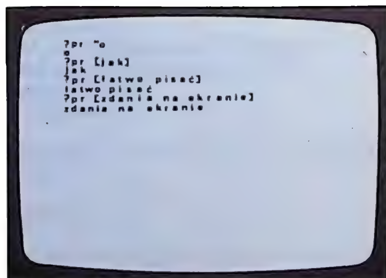
Jedną z komend wypisywania tekstów jest print lub w skrócie pr. Powoduje ona wydrukowanie wartości swego argumentu na ekranie i przejście do początku nowego wiersza. Jeśli chcemy wydrukować jakieś konkretne słowo, to należy poprzedzić je cudzysłowem, jako znakiem dosłowności. Ciąg słów należy ująć w nawiasy kwadratowe. Te reguły pozwalają wyprowadzać na ekran zarówno pojedyncze słowa, jak zdania. Na przykład (rys. 33).

```
ts
pr "o
pr [jak]
pr [łatwo pisać]
pr [zdania na ekranie]
```

Niestety Sinclair Logo nie dysponuje w swej oryginalnej wersji polskimi literami ze znakami diakrytycznymi, jak ą, ę, ć, ź itd. Mimo to używamy tu polskich liter, by nie stwarzać wrażenia sztuczności pisanych tekstów. W polskim Logo, w którym zarówno komendy, jak komunikaty przekazywane przez komputer są pisane po polsku, jest pełny polski alfabet. W wersji angielskiej też można go dorobić tak, by litery wprowadzane w trybie GRAPHICS miały polskie znaki diakrytyczne, ale tym problemem nie będziemy się tu zajmować.

Komenda setcursor lub w skrócie setcur powoduje przeniesienie wskaźnika początku tekstu drukowanego do kwadratu o podanych numerach kolumny i wiersza, ujętych w nawiasy kwadratowe. Na przykład:

```
setcur [17 7] pr [, pionowo]
setcur [27 7] pr "t
setcur [25 8] pr [ - e - ]
setcur [27 9] pr "ż
```

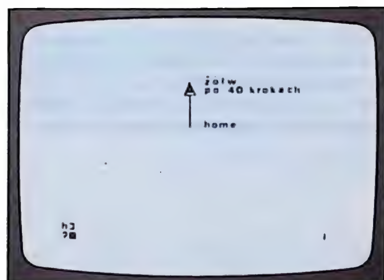


Rys. 33. Pisanie kolejnych wierszy po komendzie ts

W tym przypadku jest to sekwencja wprowadzanych komend, a nie to, co kolejno ukazuje się na ekranie. Jeśli ta sekwencja została wprowadzona po uzyskaniu na ekranie zestawu napisów jak na rysunku 33, to po jej wykonaniu powinno być:

```
?pr "o
o
?pr [jak]
jak
?pr [łatwo pisać]
łatwo pisać
?pr [zdania na ekranie]
zdania na ekranie, pionowo t
?setcur [25 8] pr [ - e - ]   - e -
?setcur [27 9] pr "ż           ż
?
```

Dalej możemy spróbować wprowadzać napisy jako uzupełnienie rysunku (rys. 34), dodając przy okazji znaki diakrytyczne tam, gdzie ich brak:



Rys. 34. Rozmieszczanie tekstów na rysunku

```
cs
setcur [18 10] pr "home
fd 40
setcur [18 4] pr "żółw
setcur [18 5] pr [po 40 krokach]
setcur [18 3] pr "..
setcur [20 4] over 1 pr "/
```

Komenda over 1 umożliwia napisanie w kwadracie znaku bez niszczenia poprzedniego, z tym że pokrywające się części obu znaków się kasują, a nie pokrywające się pozostają. Mówimy, że w kwadracie powstaje w ten sposób różnica symetryczna obu znaków, zapożyczając ten termin z rachunku zerojedynkowego: dwa zera (nie ma) lub dwie jedynki (jest) dają zero (nie ma), zaś zero i jedynka lub jedynka i zero dają jeden (jest).

Komenda over 0 powoduje powrót do stanu, gdy wpisanie nowego znaku do kwadratu jest równoznaczne ze skasowaniem starego.

Teraz możemy przejść do tworzenia kolorowych tekstów. Argumentem komendy settc jest ujęta w nawiasy kwadratowe para liczb, z których pierwsza jest numerem koloru tła tekstu (nie mylić z kolorem tła rysunku!), druga jest numerem koloru liter (lub innych znaków). Po wykonaniu tej komendy zmienia swój kolor litera w prawym dolnym rogu ekranu, oznaczająca stan klawiatury, oraz kwadrat, w którym ona się znajduje. Na początku są to jedyne objawy wykonania tej komendy. Ale każdy ciąg znaków, słowo lub zdanie pisane na ekranie przy pomocy dowolnej komendy, na przykład print, pojawia się już w tych nowych barwach:

```
settc [1 6]
setcur [5 15] pr "
setcur [5 16] pr [żółto - granatowo]
setcur [7 16] over 1 pr "/
setcur [10 18] flash pr "miga
```

Jeśli teraz zmienimy tło rysunku:

```
setbhg 5
```

to zmieni się także tło tekstów na przestrzeni rysunkowej ekranu (rys. 35). Tło



Rys. 35. To samo na innym tle i z napisami w innym kolorze

litery w prawym dolnym rogu pozostanie bez zmian. Czytelnik może zbadać skutki wielokrotnych zmian tła rysunku. Wykonanie w dowolnym momencie textscreen lub ts spowoduje skasowanie ekranu oraz nadanie mu kolorów określonych przez ostatnio wykonane settc. Powrót w tej sytuacji do trybu rysowania na ekranie oznacza powrót do koloru tła ekranu ustalonego przez ostatnią komendę setbg, koloru żółwia określonego ostatnią komendą setpc oraz koloru tła dwóch dolnych wierszy tekstu identycznego z kolorem obrzeża. Kolor litery w prawym dolnym rogu i jej własnego tła pozostaje bez zmian, jako określony przez ostatnio wykonaną settc.

Dłuższe wpatrywanie się w ciemne linie i znaki na jasnym tle jest męczące dla wzroku i należy tego unikać. Zdefiniujmy procedurę umożliwiającą wyświetlanie jasnych linii i znaków na ciemnym tle:

```
to ciemno
setbg 0 setbr 0
setpc 7 settc [0 7]
end
```

Warto mieć również procedurę o działaniu przeciwnym:

```
to jasno
setbg 7
setbr 7
setpc 0
settc [7 0]
end
```

Mając telewizor kolorowy, można utworzyć podobną procedurę dla swego ulubionego zestawu kolorów – znowu lepiej będzie wybrać tło ciemniejsze od rysunku i liter, jeśli ma się zamiar operować głównie tylko dwoma kolorami przynajmniej w pierwszej fazie pracy.

Jeśli to okaże się potrzebne, można oznaczać kolory dowolnymi liczbami całkowitymi – wtedy jako numer koloru jest brana reszta z dzielenia takiej liczby przez 8. Ale jeśli kolor jest oznaczony liczbą spoza zakresu od 0 do 7, to trudniej jest odczytać, o który chodzi. Dlatego taki niestandardowy sposób oznaczania kolorów powinno się stosować wyłącznie wtedy, kiedy to jest rzeczywiście konieczne. Może to być na przykład wtedy, gdy kolor kolejnego elementu rysunku lub tekstu jest obliczany zgodnie z jakimś wzorem.

Ale o czytelność swoich procedur i opisów stosowanych metod warto dbać zawsze.

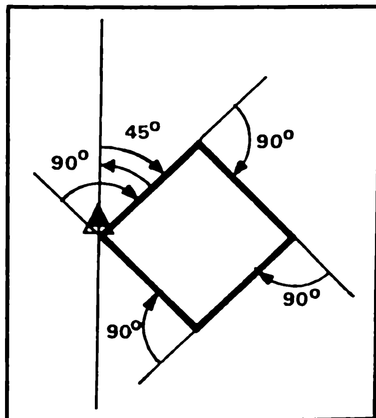
Wielokąty i okręgi

Zdefiniujmy procedurę z parametrami: rysowanie wielokąta z zadaną liczbą kątów i długością boku (rys. 36):

```
to wielokąt :ilekątów :bok
right 180 / :ilekątów
repeat :ilekątów [fd :bok rt 360 / !
:ilekątów]
lt 180 / :ilekątów
end
```

Pamiętajmy, że wykrzyknik oznacza kontynuację: wiersz Logo jest dłuższy niż wiersz tekstu drukowanego. Nie należy go pisać: pojawi się sam!

Przy okazji poznajemy przykłady użycia



Rys. 36. Zasada rysowania wielokąta na przykładzie liczby boków równej 4

operatorów arytmetycznych, w tym przypadku dzielenia. Dwukropek przed parametrem oznacza, że w tym przypadku chodzi o jego wartość – rzeczywiście w każdym miejscu bierze się aktualną wartość tego parametru, taką jaka mu została nadana w momencie wywołania procedury.

Po takim określeniu procedury wielokąt możemy użyć słowa z następującymi po nim dwoma parametrami jako komendy Logo, bądź osobno:

```
wielokąt 4 15
```

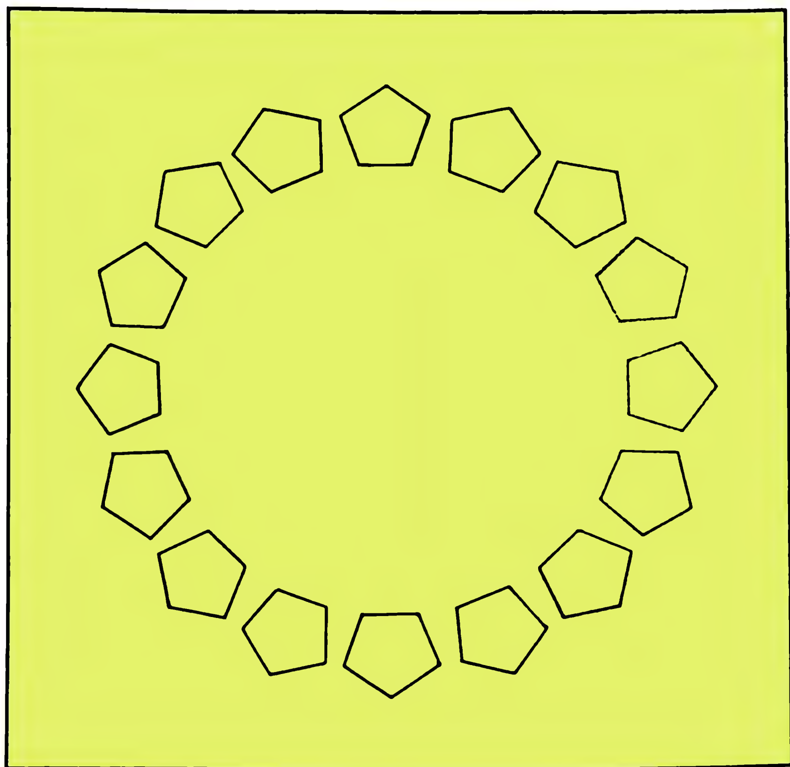
bądź też wewnątrz innej komendy

```
repeat 4 [wielokąt 5 40 rt 90]
```

lub wewnątrz innej procedury:

```
to rozeta :promień :ilepłatków
:ilekątów :bokpłatka
repeat :ilepłatków [pu fd !
:promień rt 90 pd wielokąt :ilekątów !
:bokpłatka pu lt 90 bk !
:promień rt 360 / :ilepłatków pd]
end
rozeta 80 16 5 15
rozeta 24 12 4 15
```

Jeśli wykonamy tę sekwencję komend, otrzymamy symetryczny ornament, złożony z powtarzających się elementów (rys. 37). Rysunki tego typu są interesujące i mogą się podobać dlatego, że powstają jako wynik swobodnego układania pewnych elementów w uporządkowany wzór, w którym każdy element ma swoją indywidualną, różną od innych pozycję, chociaż kształty niektórych z nich się powtarzają.



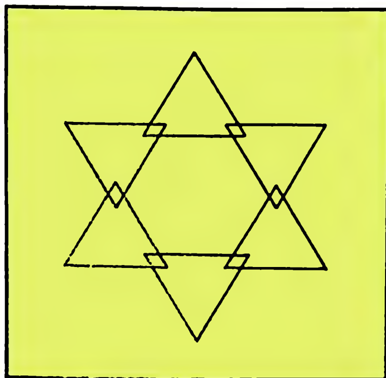
Rys. 37. Rozeta 80 16 5 15

Właśnie dobrze dobrane powtórzenia, regularności i symetria rysunku są tym, co przyciąga wzrok (rys. 38 i 39). Ale co umożliwia to swobodne komponowanie rysunków?

Przyglądając się dokładniej działaniu procedury rozeta możemy zobaczyć, że mają na nie zasadniczy wpływ następujące cechy używanej przez nią procedury wielokąt:

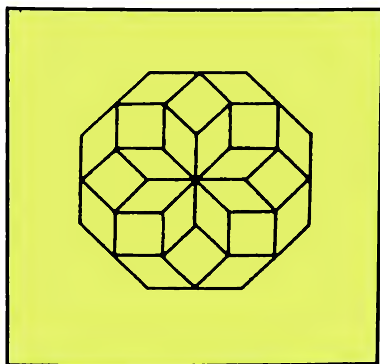
- rysuje ona wielokąt o podanej liczbie kątów i długości boku tak, że jego kształt nie zależy od jego położenia;
- położenie wielokąta na ekranie jest jednoznacznie wyznaczone przez położenie żółwia (współrzędne i kierunek) bezpośrednio przed rozpoczęciem rysowania;
- położenie żółwia bezpośrednio przed rysowaniem i bezpośrednio po nim są takie same.

O trzeciej z tych własności można powiedzieć, że położenie żółwia (współrzędne i kierunek) jest w pewnym sensie niezmiennikiem procedury: przed jej wykonaniem i po nim jest takie same. Pierwsza i druga mówią, że przesunięcia i obroty żółwia powodują przesunięcia i obroty tworzonej przez procedurę figury o taki sam wektor i kąt. Wszystko to bardzo ułatwia zadanie temu, kto chciałby z takiej procedury sko-



Rys. 38. Sześciopłatkowa rozeta z trójkątów

rzystać, bo jest on w ten sposób zwolniony od potrzeby zastanawiania się, czy rysunek po zmianie położenia nie zmieni przypadkiem swego kształtu albo jak zmieni swe położenie żółw po narysowaniu procedury. Takie jednolite podejście i stosowanie takich niezmienników uwalnia od potrzeby zwracania uwagi na różne drobiazgi techniczne związane z tworzeniem rysunku i składaniem go z różnych elementów składowych. Dzięki temu można skoncentrować uwagę na sprawach naprawdę istotnych, związanych z rozwiązywaniem zasadniczego problemu, o który właśnie w tej



Rys. 39. Rozeta z ośmiokątów

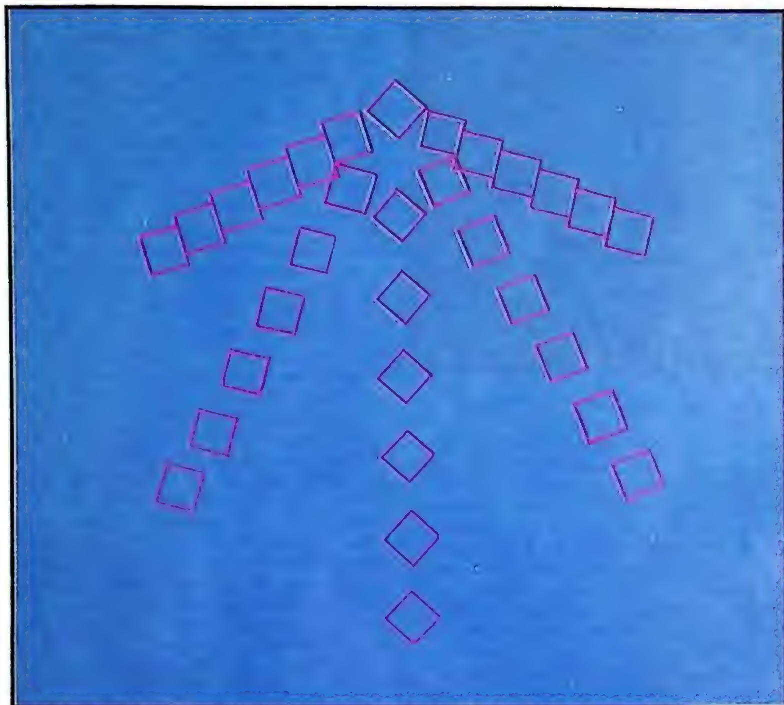
chwili chodzi. A sztuka programowania sprowadza się ostatecznie właśnie do tego, żeby móc skutecznię posługiwać się komputerem do rozwiązywania różnych zadań bez stwarzania sobie zbędnych komplikacji i utrudnień albo niepotrzebnego zaprzątania sobie głowy szczegółami, które przy prawidłowej metodzie rozwiązywania zadania w ogóle nie powinny się pojawiać. Tego, jak to osiągać, warto się uczyć od samego początku, od pierwszych kontaktów z komputerem, zwracając uwagę na czynniki, które mają znaczenie dla doboru właściwych metod pracy. Jednym z pierwszorzędnych jest przejrzysty i nie obciążony zbędnymi dodatkami sposób rozwiązywania zadania.

Pamiętający procedurę rysowania chorągiewek, korzystającą z procedury kwadrat, mogą zauważyć, że idea procedury rozeta jest podobna do tej, rysującej pęk chorągiewek: najpierw żółw siedzi w środku przyszłego rysunku, po czym wykonuje pu, przesunięcie naprzód, obrót o taki kąt, żeby móc narysować wielokąt w stronę środka rysunku, rysuje wielokąt, wraca z podniesionym pisakiem do środka rysunku, robi obrót i następnie powtarza te czynności podaną liczbę razy (por. rys. 37–39).

Zarówno rozeta, jak większość opisanych poprzednio procedur ma te same trzy własności, które były omówione dla procedury wielokąt. Zostało to zrobione celowo – dzięki temu można się nimi bardziej swobodnie posługiwać (rys. 40 i 41).

Nazwy procedur i ich parametrów powinny być zawsze tak dobierane, żeby można było łatwo wywnioskować, co dana procedura ma robić i jaką rolę mają odgrywać wartości jej parametrów. Stosowanie mało czytelnych skrótów lub oznaczeń zupełnie abstrakcyjnych i nic nie mówiących trzeba uznać za szkodliwe: po co później tracić czas na zastanawianie się, co właściwie miała robić jakaś procedura albo co miały znaczyć poszczególne parametry, gdy po pewnym czasie już nie bardzo się pamięta cały proces myślenia przy jej tworzeniu? Trzeba mieć również na uwadze to, że użytkownikiem tej procedury może być także ktoś inny, dla którego jej zapis także powinien być czytelny i zrozumiały.

Warto zwrócić uwagę na to, że przy wykonywaniu ręcznie różnych rachunków, na



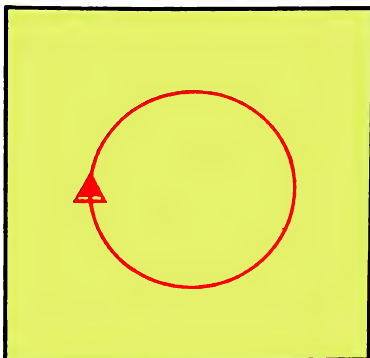
Rys. 40. Rysunek utworzony przez przesuwane wyniki procedur rozeta :promień 6 4 10

przykład w matematyce, przyzwyczajamy się do czegoś zupełnie innego. Oznaczenia zmiennych są z reguły jednoliterowe, choć mogą być opatrzone różnymi wskaźnikami, dodatkowymi znaczkami itp. Mamy do dyspozycji różne alfabety. Ręczny zapis wyraźnie nie sprzyja stosowaniu nazw zmiennych lub funkcji, które by bez żadnych wątpliwości mówiły, o co chodzi i jakie są zadania tych zmiennych lub funkcji w tym kontekście. Ale tam liczba nazw do zapamiętania jest niewielka, a wzorom towarzyszy zwykle tekst objaśniający. W programowaniu jest inaczej i dlatego przenoszenie nawyków z rachunku ręcznego nie jest zalecane.
Zwiększając liczbę kątów wielokąta, otrzymujemy twory kształtem coraz bardziej zbliżone do okręgu. Możemy to sprawdzić, pisząc:

wielokąt 36 10

– kąty są już praktycznie niewidoczne.
Przy rysowaniu okręgów wygodniej będzie posługiwać się osobną procedurą; w wielokącie pierwszy parametr byłby zawsze stały, a więc zbędny, natomiast posługiwanie się parametrem :bok do charakteryzowania wymiarów okręgu jest niewygodne i nie bardzo sensowne. Zamiast tego lepiej podać promień okręgu. Wartość parametru :bok w treści procedury można zastąpić wartością przybliżoną, obliczoną jako promień pomnożony przez liczbę pi i podzielony przez połowę trzydziestu sześciu, czyli

Rys. 41. Naszyjnik utworzony przez rozety z okręgów



Rys. 42. Wynik procedury okrąg i okrągwprawo

osiemnaście. Kąt pierwszego i ostatniego obrotu jest stały i wynosi 5 stopni.

Przyglądając się rysowaniu takiego 36-kąta widzimy, że przebiega ono dość wolno. Ale przy każdym wykonaniu ciągu komend powtarzanych przez repeat wykonuje się to samo obliczenie. Co prawda kąt obrotu jest teraz stały, ale jest obliczana wartość, odpowiadająca długości boku, zależna teraz od promienia. Obliczmy ją przed rysowaniem, wprowadzając zmienną pomocniczą :krok i nadając jej potrzebną wartość przy pomocy komendy make. Jest to komenda dwuargumentowa: pierwszym argumentem jest nazwa zmiennej, drugim nadawana wartość. Ponieważ chodzi o nazwę wziętą dosłownie, trzeba ją poprzedzić znakiem ". Ostatecznie otrzymujemy procedurę następującej postaci (por. rys. 42):

```
to okrąg :promień
make "krok :promień * 3.1415926 /!
18
rt 5
repeat 36[fd :krok rt 10]
lt 5
end
```

Redagowanie procedur: edytor

Treść dowolnej zapisanej w pamięci procedury możemy zmodyfikować, używając komendy edit lub w skrócie ed, z jednym argumentem, którego wartością jest nazwa tej procedury. Jeśli chcemy modyfikować od razu treść kilku procedur, wtedy wartością parametru komendy edit powinien być ciąg nazw tych procedur, ujęty w nawiasy kwadratowe. Różnymi wariantami wywoływania edit zajmiemy się później, na razie omówimy jej użycie w przypadku najprostszym, redagowania jednej procedury. Jako przykład może służyć świeżo zdefiniowana procedura okrąg. Napisanie:

ed "okrąg

powoduje zniknięcie tego, co było na ekranie przedtem i wypisanie, poczynając od pierwszego wiersza, pełnej treści redagowanej procedury. Na dole ekranu jest odpowiedni napis, sygnalizujący, że mamy do czynienia z pracą w trybie redagowania i że to, co jest na ekranie, jest stroną redagowanego tekstu. Oprócz tego w zwykłym miejscu, w prawym dolnym rogu jest litera oznaczająca stan klawiatury. Migający wskaźnik jest na pierwszej literze tekstu, w lewym górnym rogu ekranu.

Aby poprawić zapis, trzeba doprowadzić wskaźnik do odpowiedniego miejsca tekstu i następnie wprowadzić odpowiednie zmiany. Do przesuwania wskaźnika służą klawisze ze strzałkami. Na Spectrum z gumową klawiaturą są to klawisze w prawej górnej części klawiatury i trzeba je naciskać równocześnie z klawiszem CAPS SHIFT. Spectrum plus ma klawisze ze strzałkami w dolnej części klawiatury i są one samodzielne – naciskanie CAPS SHIFT nie jest potrzebne, chociaż nie przeszkadza (rys. 43).

Samo przesuwanie wskaźnika żadnych zmian tekstu nie powoduje i wobec tego można zbadać eksperymentalnie działanie klawiszy ze strzałkami w różnych sytuacjach, jak: dojście do końca lub początku wiersza, początku lub końca tekstu itp. Poprawiany tekst jest automatycznie podzielony na strony i może być przewijany (jeśli jest dostatecznie długi) tak, że można oglądać kolejne wiersze długiego tekstu i stronę po stronie tylko operując odpowiednio przesuwaniem wskaźnika. Ale na razie nasz tekst mieści się na jednej stronie i te efekty będziemy mogli zbadać później.

Jeszcze dwa klawisze, używane przy redagowaniu, są inne w obu wersjach Spectrum: DELETE i BREAK. W Spectrum z gumową klawiaturą wymagają one jednoczesnego naciśnięcia CAPS SHIFT, w Spectrum plus są samodzielne. Jak w przypadku strzałek naciśnięcie CAPS SHIFT nie jest potrzebne, chociaż nie szkodzi.

Naciśnięcie BREAK powoduje przerwanie redagowania bez uwzględniania poprawek. Jest to więc klawisz, którego raczej nie należy naciskać przez pomyłkę. Tekst, który miał być zredagowany, pozostaje w takim stanie, w jakim był przed wywołaniem komendy edit.

DELETE powoduje usunięcie jednego znaku na lewo od wskaźnika. I dalej:

EXTENDED MODE ← przerzuć wskaźnik na początek wiersza.

EXTENDED MODE → przerzuć wskaźnik na koniec wiersza.

EXTENDED MODE ↑ przerzuć wskaźnik na początek ekranu.

EXTENDED MODE ↓ przerzuć wskaźnik na

Spectrum +	Spectrum 48k i Spectrum +	
	przytrzymać	nacisnąć
DELETE	CAPS SHIFT	Ø
GRAPH	CAPS SHIFT	9
⇨	CAPS SHIFT	8
⇧	CAPS SHIFT	7
⇩	CAPS SHIFT	6
⇦	CAPS SHIFT	5
INV VIDEO	CAPS SHIFT	4
TRUE VIDEO	CAPS SHIFT	3
CAPS LOCK	CAPS SHIFT	2
EDIT	CAPS SHIFT	1
EXTEND MODE	CAPS SHIFT	SYMBOL SHIFT
BREAK	CAPS SHIFT	SPACE
;	SYMBOL SHIFT	O
"	SYMBOL SHIFT	P
,	SYMBOL SHIFT	N
.	SYMBOL SHIFT	M

Rys. 43. Wykonanie na obu typach Spectrum czynności powodujących te same rezultaty co naciskanie klawiszy istniejących tylko na Spectrum +

koniec ekranu albo na koniec tekstu, jeśli ten jest na ekranie.
EXTENDED MODE B przerzuć wskaźnik na początek tekstu.
EXTENDED MODE E przerzuć wskaźnik na koniec tekstu.
EXTENDED MODE P przerzuć wskaźnik na poprzednią stronę.
EXTENDED MODE N przerzuć wskaźnik na następną stronę.
SYMBOL SHIFT S wstrzymaj przewijanie. Po naciśnięciu dowolnego klawisza przewijanie tekstu zostanie wznowione.
EXTENDED MODE Y usuń część wiersza na prawo od wskaźnika.
EXTENDED MODE R wstaw tam, gdzie jest wskaźnik, to, co zostało ostatnio usunięte przez EXTENDED MODE Y.
EXTENDED MODE C zakończ redagowanie i zapamiętaj poprawiony tekst.
EXTENDED MODE Z wyłącz (lub włącz, jeśli jest wyłączony) dźwięk wydawany przy naciskaniu klawisza i po zakończeniu wykonania procedury.

Jak pamiętamy, w Spectrum z gumową klawiaturą przechodzi się do trybu EXTENDED MODE, naciskając CAPS SHIFT i SYMBOL SHIFT. Na Spectrum plus można zrobić to samo, ale istnieje też specjalny klawisz sprowadzający do tego trybu. W prawym dolnym rogu pojawia się litera E (co warto sprawdzić). Po naciśnięciu dowolnego klawisza następuje powrót do poprzedniego stanu klawiatury.
Naciśnięcie dowolnego klawisza znakowego powoduje wpisanie odpowiedniego znaku bezpośrednio na lewo od wskaźnika. Spróbujmy zastosować to w naszym przykładzie.
Mając wskaźnik na pierwszej literze tekstu przesuwamy go (strzałkami w prawo) na odstęp po literze G i piszemy:

wprawo

korygując w ten sposób nazwę procedury. EXTENDED MODE C powoduje zapisanie jej pod nową nazwą i zakończenie redago-

wania. Na ekranie w trybie tekstowym ukazuje się napis:

OKRĄGWPRAWO defined

i poniżej pytajnik z migającym wskaźnikiem. Ponieważ nazwa procedury została zmieniona, stara procedura, okrąg, jest także pamiętana.

Na razie poprawka posłużyła tylko do wyraźnego zaznaczenia faktu, że rysowany okrąg jest styczny do prostej, na której siedzi żółw, w punkcie, w którym siedzi, i że rysowanie okręgu zaczyna się w prawo. Podobnie można zdefiniować procedurę rysowania okręgu w lewo. Zamiast pisać ją od nowa, łatwiej będzie ją uzyskać przez redagowanie istniejącej:

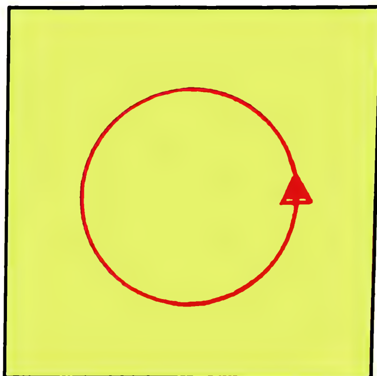
edit okrągwprawo

Znak dosłowności w argumencie tej komendy można opuszczać – jest to jej wyjątkowa cecha, na ogół nie spotykana w innych komendach. Jeżeli bierze się do redagowania ten sam tekst procedury, który był już redagowany bezpośrednio przedtem, to argument edit można w ogóle opuścić. Inaczej mówiąc, użycie edit lub ed bez żadnego argumentu oznacza, że wartość argumentu ma być taka sama, jak w poprzednim wywołaniu tej komendy.

Sprowadźmy wskaźnik na miejsce bezpośrednio po prawej stronie liter pra i trzema naciśnięciami DELETE skasujmy tę trójkę, a następnie, nie ruszając wskaźnika, napiszmy le. Dalej zamieńmy wszystkie r na l i odwrotnie we wszystkich rt i lt. EXTENDED MODE C kończy redagowanie tej procedury, która ostatecznie powinna wyglądać tak (por. rys. 44):

```
to okrągwlewo :promień
make "krok :promień * 3.1415926 / !
18
lt 5
repeat 36 [fd :krok lt 10]
rt 5
end
```

Aby przećwiczyć redagowanie dwu procedur jednocześnie, zdefiniujmy sobie w podobny sposób, przez redagowanie, procedury ćwierćokrągwprawo i ćwierćokrągwlewo. Tym razem argumentem procedury są dwie nazwy w nawiasach kwadratowych, których nie można opuścić:



Rys. 44. Wynik procedury okrągwlewo

ed [okrągwprawo okrągwlewo]

Dla każdej procedury dopisujemy ćwierć w nazwie i zastępujemy 36 przez 9. Po EXTENDED MODE C na ekranie pojawia się napis:

ĆWIERĆOKRĄGWPRAWO defined
ĆWIERĆOKRĄGWLEWO defined

W procedurze rozeta mamy w każdym kroku dwa przejścia żółwia z podniesionym pisakiem: przed narysowaniem kolejnego elementu rozety i po. Określmy sobie w zwykły sposób procedurę, pisząc:

```
to przeskocz :odległość
pu fd :odległość pd
end
```

Teraz możemy spróbować rysować także obrazki, jak na rys. 45–47. Wprowadzenie tej ostatniej procedury pozwoli też poprawić procedurę rozeta tak, aby te dwa przejścia były opisane przy pomocy wywołań procedury przeskocz. Wymaga to innego zgrupowania komend, ale skorzystamy tu z tego, że pu i pd są przestawialne z obrotami oraz że fd z ujemnym argumen-

tem działa jak bk. Dojście do formy poprawionej:

```
to rozeta :promień :ilepłatków !
:ilekątów :bokpłatka
repeat :ilepłatków [przeskocz !
:promień rt 90 wielokąt :ilekątów !
:bokpłatka lt 90 przeskocz !
-:promień rt 360 / :ilepłatków]
end
```

może być jeszcze jedną okazją przećwiczenia sprawnego wprowadzania drobnych korekt – tym razem nieco bardziej skomplikowanych, niż poprzednie. Ponieważ poprawiona procedura nie zmienia nazwy, po EXTENDED MODE C będzie zapisana nowa wersja na miejsce starej, która nie będzie zachowana.



Rys. 45. Okręgi i ćwierćokręgi

Użycie EXTENDED MODE Y do usuwania większych fragmentów tekstu jest dość oczywiste. Zważywszy, że odpowiednie użycie DELETE i ENTER pozwala dość dowolnie łączyć i rozdzielać wiersze, wypreparowanie fragmentów usuwanych w taki sposób z praktycznie każdego kontekstu nie powinno nastręczać trudności. Ale ta operacja w połączeniu z EXTENDED MODE R pozwala także przenosić albo kopiować takie fragmenty.
Komenda ed, której argumentem jest nazwa nie istniejącej procedury, powoduje pojawienie się strony edytora, w której jest tylko ta nazwa, poprzedzona TO. Stosując normalne operacje redagujące można teraz



Rys. 46. Tu ćwierćokręgi się przydadzą

ten napis uzupełnić tak, aby powstała cała procedura. Gdybyśmy zresztą chcieli, można by w ten sposób zredagować dowolny tekst – Logo nie sprawdza po zakończeniu redagowania, czy jego wyni-



Rys. 47. Jak to narysować?

kiem jest właśnie poprawnie zbudowana procedura. Dzięki temu można użyć edytora także do tworzenia dowolnych danych tekstowych.
Utwórzmy w ten sposób procedurę cyrkiel, rysującą okrąg o zadanym promieniu tak, że początkowe położenie żółwia wyznacza jego środek (rys. 48). Żółw rysuje okrąg w prawo, po czym wraca do poprzedniego położenia w środku. Po napisaniu:

ed cyrkiel

ukazuje się na ekranie strona edytora z napisanym w górnym wierszu:

TO CYRKIEL



Rys. 48. Wynik procedury cyrkiel

i migającym wskaźnikiem na literze T. EXTENDED MODE → sprowadza wskaźnik na prawo od litery L, po czym dopisujemy odstęp oraz :promień. ENTER przerzuca wskaźnik na początek następnego wiersza, w którym piszemy:

przeskocz :promień

(tym razem bez ENTER!). EXTENDED MODE ← sprowadza wskaźnik na początek wiersza, po czym piszemy EXTENDED MODE Y i EXTENDED MODE R. Z pozoru nic się nie stało, ale w ten sposób ten wiersz został zanotowany do skopiowania. Teraz strzałka ↓ przenosi wskaźnik na początek następnego wiersza, w którym piszemy.

rt 90 okrąg :promień lt 90 ENTER

Po EXTENDED MODE R pojawia się w kolejnym wierszu:

przeskocz :promień

i strzałką → przesuwamy wskaźnik tak, aby umieścić znak – bezpośrednio przed parametrem, otrzymując:

przeskocz :promień

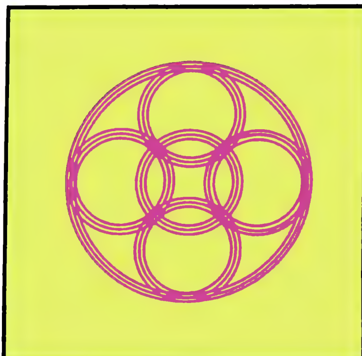
Dalej piszemy kolejno:

EXTENDED MODE → ENTER
end ENTER
EXTENDED MODE C

i na ekranie w trybie tekstowym ukazuje się:

CYRKIEL defined

Teraz możemy narysować motyw architektoniczny (rys. 49).Jako inny przykład zastosowania takiego sposobu weźmy rysowanie pięciu okręgów o zadanym promieniu, ułożonych tak, jak kółka olimpijskie, z tym, że sąsiadujące bezpośrednio ze sobą są w



Rys. 49. Można tu użyć procedury cyrkiel albo rozeta

pewnej zadanej odległości. Załóżmy dodatkowo, że środek centralnego okręgu jest wyznaczony przez początkowe położenie żółwia. Jeśli zdecydujemy się użyć do rysowania procedury cyrkiel, to jedynym problemem jest zaplanowanie wędrówki żółwia po łamanej, której wierzchołkami są środki kolejno rysowanych okręgów (rys. 50). Jak widać, odcinki tej łamanej są stałe, a kąty zwrotów żółwia w kolejnych wierzchołkach pokazuje rysunek. Można napisać procedurę tak, by miała pięć kroków, różniących się tylko argumentem komendy obrotu żółwia, oraz na końcu dodatkowy obrót, sprowadzający żółwia do pozycji wyjściowej:

ed kółkaolimpijskie

powoduje pojawienie się na ekranie nazwy procedury, do której dopisujemy z prawej nazwy parametrów, poprzedzone dwukropkami, otrzymując:

TO KÓŁKAOLIMPIJSKIE :promień :odległość

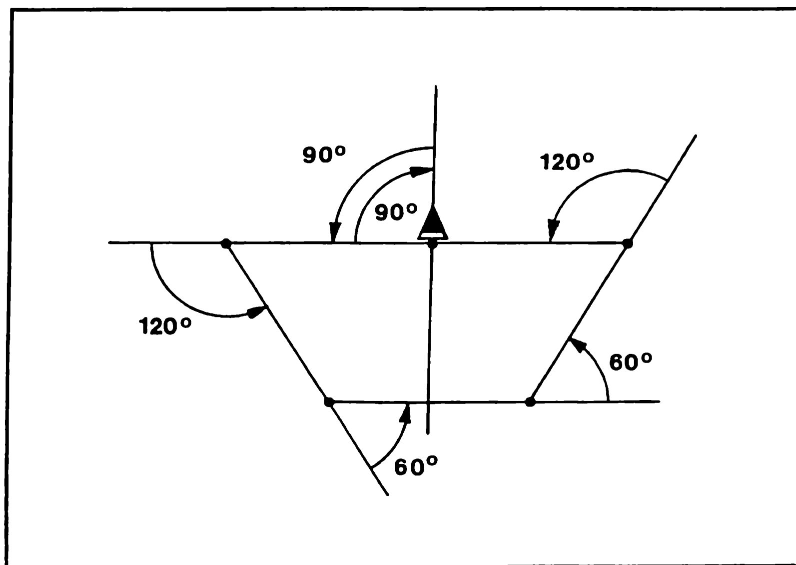
W następnym wierszu piszemy (bez ENTERI)

lt 90 przeskocz :odległość cyrkiel ! :promień

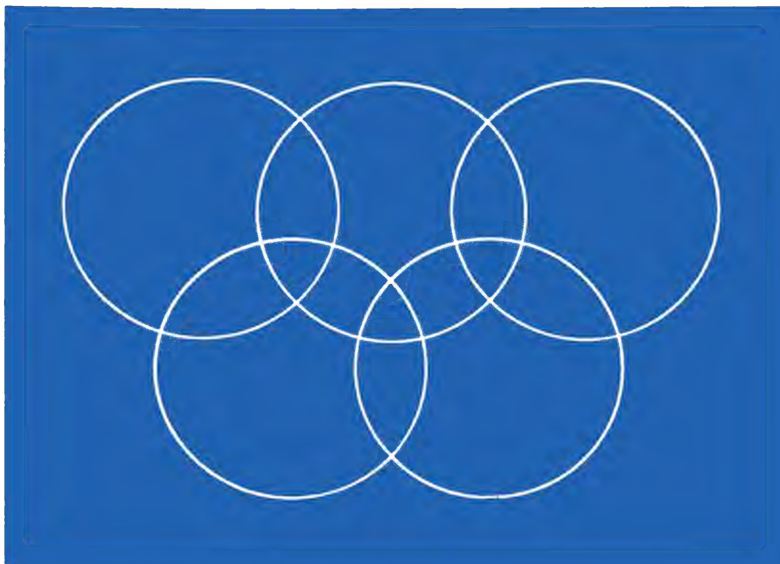
i dalej EXTENDED MODE ←, strzałkę doprowadzającą do początku tego ciągu komend, EXTENDED MODE Y, EXTENDED MODE R i czterokrotnie po dwie strzałki ↓ oraz EXTENDED MODE R. Tak powstały ciąg wierszy uzupełniamy:

rt 90
end

i pozostają tylko do poprawienia niektóre kąty w komendach lt, zgodnie z rysunkiem. Ostatecznie powinniśmy otrzymać (por.rys. 51):



Rys. 50. Planowana trasa żółwia

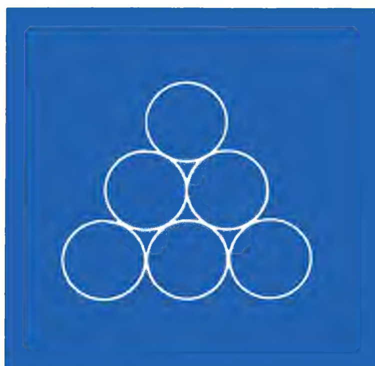


Rys. 51. Kółkaolimpijskie 10 15

```
TO KÓŁKAOLIMPIJSKIE :promień !
:odległość
lt 90 przeskocz :odległość cyrkiel !
:promień  lt 120 przeskocz :odległość
cyrkiel ! :promień
lt 60 przeskocz :odległość cyrkiel !
:promień
lt 60 przeskocz :odległość cyrkiel !
:promień
lt 120 przeskocz :odległość cyrkiel !
:promień
rt 90
end
```

Teraz możemy zobaczyć co się stanie, gdy na przykład napiszemy:

```
cs lt 60 repeat 7 [kółkaolimpijskie 10 !
15 przeskocz 45]
```



Rys. 52. Tak też można

Nieco inaczej zaprogramujemy tworzenie rys. 52.
Czytelnik może sam skrócić zapis tej procedury, wprowadzając repeat. Przy takiej modyfikacji znowu może się przydać EXTENDED MODE Y, tym razem do usuwania całych wierszy.
Ostatnim wreszcie wariantem wywołania

komendy edit jest użycie jej z pustą listą nazw:

ed []

co powoduje pojawienie się pustej strony edytora, zawierającej tylko standardowy napis na dole ekranu. Aby napisany tekst mógł być prawidłowo zapamiętany po EXTENDED MODE C, jego pierwszy i ostatni wiersz muszą mieć postać wymaganą dla procedur. Ostatni wiersz powinien więc zawierać END, zaś pierwszy prawidłowo zbudowany nagłówek procedury: kolejno to, nazwę oraz ewentualnie parametry. Tekst pomiędzy tymi dwoma wierszami może być całkowicie dowolny, o ile oczywiście nie zechcemy posłużyć się nazwą z nagłówka w komendach Logo. Pod tym względem dla tego wariantu obowiązują dokładnie takie same zasady, jak dla innych wersji wywołań edytora.

Redagowanie wierszy

Poza trybem redagowania, to znaczy gdy na początku wiersza jest ? albo >, można używać prawie wszystkich klawiszy do poprawek jak w edytorze, ale ich działanie w chwili użycia ogranicza się do jednego wiersza Logo. Z listy podanej wyżej dla edytora należy wykluczyć użycie po EXTENDED MODE klawiszy C i N. B i ↑ oraz E i ↓ po EXTENDED MODE działają parami jednakowo: przerzucają wskaźnik odpowiednio na początek lub na koniec wiersza Logo, podczas gdy EXTENDED MODE ← i → działają, tak samo jak w edytorze, w ramach wiersza ekranu. EXTENDED MODE P również przerzuca wskaźnik na początek wiersza Logo. Y po EXTENDED MODE usuwa część wiersza Logo na prawo od wskaźnika, zaś R powoduje wpisanie na miejscu, na którym stoi wskaźnik, tego co zostało usunięte przez EXTENDED MODE Y, albo wiersza wprowadzonego ostatnio przez ENTER, zależnie od tego, co nastąpiło później. Klawisze → , ← , DELETE oraz wszystkie znakowe użyte w zwykły sposób działają jak w edytorze, dopóki wskaźnik mieści się w ramach wiersza Logo, a długość tego wiersza nie przekracza 242 znaków. Próby przekroczenia tej liczby znaków są nieskuteczne. ENTER powoduje zawsze wczytanie wiersza, niezależnie od tego, na którym miejscu wiersza znajduje się wskaźnik. Jest to wyraźna różnica w porównaniu z edytorem, gdzie klawisz ENTER mógł służyć tylko do przechodzenia do nowego wiersza lub łamania istniejących. Podobnie DELETE nie może służyć do łączenia wierszy Logo, jak to było w edytorze, gdyż próba wyjścia do wiersza poprzedniego, nawet gdyby on jeszcze był na ekranie, byłaby nieskuteczna.

Operowanie EXTENDED MODE R może być dość pożyteczne i warto mu poświęcić trochę uwagi. Oto przykłady:

kolkalimpijskie 32

spowoduje sygnalizację błędu:

Not enough inputs to KÓŁKAOLIMPIJSKIE

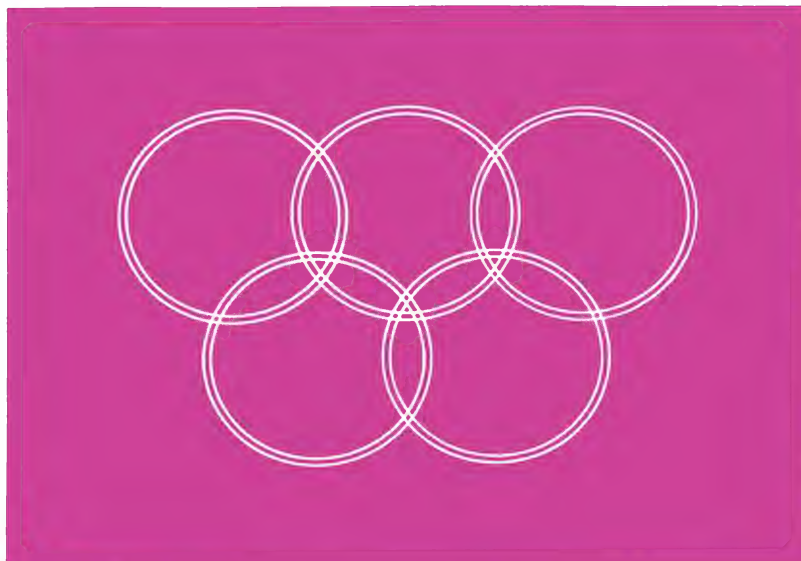
(co oznacza: Za mało danych dla KÓŁKAOLIMPIJSKIE). Rzeczywiście, brak drugiego argumentu. EXTENDED MODE R sprowadza kopię tego wiersza Logo. Wskaźnik jest na pierwszej literze. Napiszmy cs z odstępem (aby oczyścić ekran przed rysowaniem) i po EXTENDED MODE E (albo→, wszystko jedno) dopisujemy odstęp i liczbę 50. Potem ENTER. Po narysowaniu tych kółek możemy wywołać podobną komendę, tworząc ją z poprzedniego wiersza Logo. Sprowadzamy go przez EXTENDED MODE R (bo teraz można w ten sposób otrzymać tylko to, co zostało właśnie ostatnio wprowadzone przez ENTER) i usuwając cs oraz zmieniając pierwszy argument otrzymujemy (rys. 53):

kółkaolimpijskie 30 50

Zajmijmy się teraz innym ciągiem komend:

cs lt 90 przeskocz 120 rt 135
ćwierćokręgwprawo 15

Wykonanie go tworzy łuk po lewej stronie ekranu. Możemy go teraz zacząć rozbudowywać, żeby robił więcej. EXTENDED MODE R sprowadza znów ten wiersz Logo



Rys. 53. Kółkaolimpijskie w odległości 50 z promieniami 30 i 32

na ekran, zaś EXTENDED MODE E (ale nie strzałka →, bo chodzi o dojście do końca wiersza Logo, a nie wiersza ekranu) sprowadza wskaźnik za liczbę 15. Kolejno wykonane EXTENDED MODE ←, EXTENDED MODE Y, EXTENDED MODE R, EXTENDED MODE →, napisanie odstępu i EXTENDED MODE R dają ten wiersz ze skopiowaną ostatnią komendą. Zamiana ostatniego prawo na lewo jest już łatwe i otrzymaliśmy w ten sposób ciąg komend, który przy brzegu ekranu rysuje całą falkę. Można go dalej przekształcać, dając repeat 5 [przed pierwszym z ćwierćokręgów i nawias] na końcu (po użyciu EXTENDED MODE E, rys. 54).

Widać tu zarówno przykłady kopiowania już wykonanych ciągów komend Logo, jak rozszerzania redagowanego wiersza Logo przez kopiowanie jego fragmentów. W



Rys. 54. Rysowanie fali

połączeniu z pokazanym poprzednio poprawianiem komend po błędzie stwarza to rzeczywiście dość dogodne możliwości działania dla wprawnego programisty, ułatwiając wyjście z różnych trudnych lub kłopotliwych sytuacji.
Ciągi komend, pisane tak, jak w ostatnich przykładach, są ulotne – już wprowadzenie następnego, innego wiersza Logo przy pomocy ENTER powoduje ich bezpowrotną utratę. Zanim jednak to nastąpi, można taki wiersz Logo przekształcić w procedurę. Posłużymy się w tym celu procedurą define. Nie będziemy tu opisywać wszystkich jej możliwości ani zastosowań, poprzestając na tym jednym.
Jest to procedura dwuargumentowa. Wartością pierwszego argumentu jest nazwa nadawana nowo tworzonej procedurze, drugiego jest treść tej nowej procedury, ujęta w nawiasy kwadratowe w dość specjalny sposób. Prześledźmy na przykładzie, jak to się robi.
Sprowadzamy przy pomocy EXTENDED MODE R interesujący nas wiersz Logo – w tym przypadku jest to rysowanie fal (rys. 54). Wskaźnik jest na początku wiersza i wobec tego możemy od razu napisać sekwencję, która ma ten wiersz poprzedzić:

define " fala [[][

po czym przerzucając wskaźnik na koniec wiersza Logo przy pomocy EXTENDED MODE E dopisujemy tam dwa nawiasy zamykające]]. Naciśnięcie ENTER powoduje zarejestrowanie tego wiersza jako procedury bezparametrowej. Pisząc:

fala

możemy się przekonać, że działa, natomiast:

ed fala

pozwala przyjrzeć się, co z tego wynikło, a także zmodyfikować procedurę przez pocięcie tego ciągu komend na bardziej czytelne krótsze wiersze, lub w ogóle zmodyfikowanie treści tej procedury. W szczególności warto by się zatroszczyć o to, by po jej wykonaniu sprowadzać żółwia z powrotem do pozycji, w której był poprzednio, dzięki czemu będzie łatwo wykorzystać tę procedurę do rysowania ciągu fal. Wystarczy w tym celu dopisać przed END jeszcze jeden wiersz:

lt 135 przeskocz 86 + random 10 rt 90

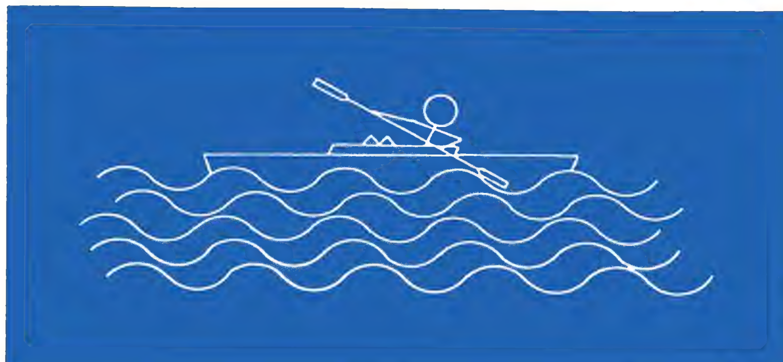
Żółw zostaje ustawiony pionowo, ale nie wraca dokładnie na to samo miejsce, na którym był poprzednio – jego pozycja jest dobierana losowo. Ma jednak dobre położenie do tego, aby można było bezpośrednio po tym zacząć rysować drugą falę, a o to właśnie chodziło. Aby można było mieć na rysunku więcej takich fal, usuńmy cs z początku treści tej procedury. Teraz możemy stworzyć nową procedurę:

to fale :ilefal
repeat :ilefal [fala przeskocz 10]
end
fale 9

Widać, że kolejne fale są nieco poprzesuwane względem siebie (rys. 55). Przesunięcia się zresztą kumulują, co wynika ze sposobu ich powstawania. Można uznać,



Rys. 55. Fale

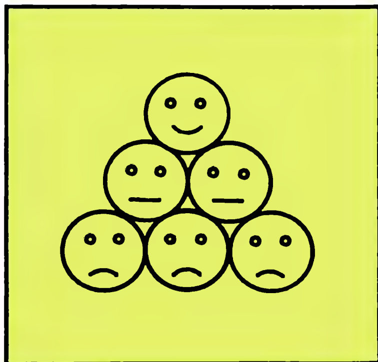


Rys. 56. Jak to narysować?

że właśnie tak fale powinny wyglądać. Gdybyśmy jednak uznali, że wszystkie początki fal powinny się zaczynać w miejscach o tych samych współrzędnych x, to trzeba by dokładniej ustalić, gdzie wypadają końce fal. Fala składa się z 10 ćwierćokręgów o promieniu 15, a więc współrzędna jej końca będzie równa 300*(sin 45) - 120 i w takiej właśnie pisowni można to przedstawić w Logo. Możemy to wyrażenie podstawić jako argument ostatniej komendy przeskocz w procedurze fala. Po tej poprawce ta procedura będzie wyglądać tak:

```
to fala
lt 90 przeskocz 120 rt 135
repeat 5 [ ćwierćokrągwprawo !
15 ćwierćokrągwlewo 15]
lt 135 przeskocz 300  * (sin 45) - 120 !
rt 90
end
```

Nawiasów, obejmujących sin 45 nie można opuścić, gdyż w Logo pierwszeństwo mają zawsze operacje *, /, +, - przed operacjami opisanymi przy pomocy nazw literowych, jak właśnie sin, a także cos, tan, cot, sqrt, które to nazwy oznaczają kolejno cosinus, tangens, cotangens, pierwiastek kwadratowy. Szczegółowo o tych wszystkich funkcjach niżej. Nie można również opuścić odstępu pomiędzy sin a 45, gdyż sin jest nazwą funkcji trygonometrycznej, znaną Logo, zaś sin45 też byłaby potraktowana jako nazwa, której jednak trudno przypisać jakiś sens - jej pojawienie się spowodowałoby zasygnalizowanie błędu, jak zwykle w przypadku, gdy argument nie jest prawidłowo poprzedzony odstępem.
Procedura fale pozwala sprawdzić, jaki jest rezultat tej poprawki.
Należy mieć nadzieję, że dotychczasowy wykład pozwoli Czytelnikowi sprostać zadaniom z rys. 56 i 57. Jeżeli tak, to może już on przystąpić do studiowania drugiej części tej książki.



Rys. 57. Kółka olimpijskie jeszcze raz

Wykaz komend Logo

Ten rozdział zawiera spis wszystkich komend języka Logo (wersji angielskiej) z krótkimi objaśnieniami. Słowa op przy nazwie komendy oznaczają operację, to znaczy taką procedurę języka, przy której wykonaniu jest obliczana wartość, przypisywana nazwie tej procedury. Dla każdej komendy jest również podawana liczba argumentów; jeśli jest ona zmienna, to podana jest najmniejszą możliwą.
Dla komend nie wymienionych w poprzednich rozdziałach może nie być jasne ich przeznaczenie i sposób użycia, ale większość z nich będzie omówiona w drugiej części tej książki.

and	op	2	logiczne i
arccos	op	1	arccos
arccot	op	1	arcctg
arccotangent	op	1	arcctg
arcsin	op	1	arcsin
arctan	op	1	arctg
arctangent1	op	1	arctg
ascii	op	1	nr znaku w kodzie ASCII
back		1	przesuń żółwia wstecz
background	op	1	numer koloru tła rysunku
bf	op	1	lista lub słowo bez pierwszego elementu
bg	op	1	numer koloru tła rysunku
bk		1	przesuń żółwia wstecz
bl	op	1	lista lub słowo bez ostatniego elementu
bright		1	pisz jaskrawo (lub normalnie)
butfirst	op	1	lista lub słowo bez pierwszego elementu
butlast	op	1	lista lub słowo bez ostatniego elementu
bye		0	koniec pracy Logo
catalog		0	katalog dyskietki lub microdrive'u
char	op	1	znak o podanym kodzie ASCII
clean		0	zmaż ekran, nie ruszając żółwia
clearscreen		0	czyść ekran
cleartext		0	zmaż tekst z całego ekranu
copydef		2	powiel definicję procedury
copyscreen		0	drukuj obraz z ekranu na drukarce
cos	op	1	cos
cosine	op	1	cos
cot	op	1	ctg
cotangent	op	1	ctg
count	op	1	długość słowa lub listy
cs		0	czyść ekran
ct		0	zmaż tekst z całego ekranu

cursor	op 0	położenie wskaźnika
define	2	określ procedurę
definedp	op 1	czy procedura jest określona?
div	op 2	iloraz
dot	1	rysuj punkt
ed	1	redaguj
edit	1	redaguj
edns	1	redaguj wszystkie nazwane tu zmienne
emptyp	op 1	czy puste słowo lub lista?
end	0	zakończenie definicji procedury
equalp	op 2	czy równe?
er	1	usuń procedurę
erall	0	usuń wszystko
erase	1	usuń procedurę
erasefile	1	usuń plik z dyskietki lub microdrive'u
ern	1	usuń wszystkie nazwane tu zmienne
erns	0	usuń wszystkie zmienne
erps	0	usuń wszystkie procedury
false	0	fałsz (jako stała logiczna)
fd	1	przesuń żółwia naprzód
fence	0	postaw płot wokół ekranu
first	op 1	pierwszy element listy lub słowa
flash	0	pisz migający napis
forward	1	przesuń żółwia naprzód
fput	op 2	wstaw pierwszy element
heading	op 0	kąt położenia żółwia
hideturtle	0	schowaj żółwia
home	0	wróć żółwiem do początku układu
ht	0	schowaj żółwia
if	2	jeśli
int	op 1	część całkowita liczby
inverse	0	pisz napis odwracając kolory (negatyw)
item	op 2	element listy lub słowa
keyp	op 0	czy był naciśnięty klawisz?
last	op 1	ostatni element listy lub słowa
left	1	obróć żółwia w lewo
list	op 2	stwórz listę
listp	op 1	czy jest listą?
load	1	ładuj z pamięci zewnętrznej
loadd	1	ładuj redagowane procedury lub dane
loadscr	1	ładuj obraz z pamięci zewnętrznej
lput	op 2	wstaw ostatni element
lt	1	obróć żółwia w lewo
make	2	przypisz zmiennej podaną wartość
memberp	op 2	czy jest elementem listy lub słowa?
name	2	nazwij
namep	1	czy jest nazwą zmiennej?
nodes	op 0	liczba wolnych miejsc pamięci (w sensie Logo)
normal	0	pisz napis bez odwracania kolorów (pozytyw)
not	op 1	logiczne nie
numberp	op 1	czy jest liczbą?
op	1	wynik operacji
or	op 2	logiczne lub
output	1	wynik operacji

over		1	pisz napis, nadrukowując na poprzednim
pc	op	0	kolor pisaka
pd		0	opuść pisak
pe		0	włącz ścieranie
pencolour	op	0	kolor pisaka
pendown		0	opuść pisak
penerase		0	włącz ścieranie
penreverse		0	włącz odwracanie kreski
penup		0	podnieś pisak
po		1	pokaż procedurę
poall		0	pokaż wszystkie procedury i zmienne
pons		0	pokaż wszystkie zmienne
pops		0	pokaż wszystkie procedury
pos	op	0	pozycja żółwia
position	op	0	pozycja żółwia
pots		0	pokaż tytuły wszystkich procedur
pr		1	pisz na ekranie
primitivep	op	1	czy jest nazwą procedury pierwotnej?
print		1	pisz na ekranie
printoff		0	wyłącz druk na drukarce
printon		0	włącz druk na drukarce
product	op	2	iloczyn
pu		0	podnieś pisak
px		0	włącz odwracanie kreski
random	op	1	losuj liczbę
rc	op	0	czytaj znak z klawiatury
readchar	op	0	czytaj znak z klawiatury
readlist	op	0	czytaj listę
recycle		0	odśmieć pamięć
remainder	op	2	reszta z dzielenia
repeat		2	powtórz
right		1	obróć żółwia w prawo
rl		0	czytaj listę
round	op	1	zaokrąglij liczbę
rt		1	obróć żółwia w prawo
run	op	1	zrób to, co jest podane w argumencie
save		2	zapisz w pamięci zewnętrznej
saveall		1	zapisz wszystko w pamięci zewnętrznej
saved		1	zapisz redagowane procedury lub dane
savescr		1	zapisz obraz w pamięci zewnętrznej
scrunch	op	0	proporcja skali osi y do x
se	op	2	stwórz listę z elementów argumentów
sentence	op	2	stwórz listę z elementów argumentów
setbg		1	określ kolor tła rysunku
setborder		1	określ kolor ramki (obrzeża) ekranu
setbr		1	określ kolor ramki (obrzeża) ekranu
setcur		1	ustaw wskaźnik (kursor)
setcursor		1	ustaw wskaźnik (kursor)
setdrive		1	wybierz dyskietkę, microdrive lub magnetofon
seth		1	ustaw żółwia pod podanym kątem
setheading		1	ustaw żółwia pod podanym kątem
setpc		1	określ kolor pisaka
setpos		1	przesuń żółwia na podaną pozycję
setscr		1	określ proporcje skali osi y do x

setscrunch		1	określ proporcje skali osi y do x
settc		1	określ kolory tekstu (tła i znaków)
setx		1	przesuń żółwia poziomo
sety		1	przesuń żółwia pionowo
show		1	pokaż listę lub słowo
shownp	op	0	czy żółw jest widoczny?
showturtle		0	pokaż żółwia
sin	op	1	sin
sine	op	1	sin
sound		1	graj podaną nutę
sqrt	op	1	pierwiastek kwadratowy
st		0	pokaż żółwia
startrobot		0	uruchom robota
stop		0	stop (zakończenie wykonania procedury)
stoprobot		0	zatrzymaj robota
sum		2	suma
tan	op	1	tg
tangent	op	1	tg
tc	op	0	kolory tekstu
text	op	1	treść procedury
textcolour	op	0	kolory tekstu
textscreen		0	przejście do trybu pisania tekstów
thing	op	1	wartość zmiennej
to		1	początek definicji procedury
toplevel		0	przerwij rekurencję
towards	op	0	kąt widzenia punktu przez żółwia (azymut)
true	op	0	prawda (jako stała logiczna)
ts		0	przejście do trybu pisania tekstów
type		1	wpisz tekst, nie zmieniając wiersza
wait		1	czekaj
window		0	określ ekran jako okno na płaszczyznę
word	op	2	stwórz słowo
wordp	op	1	czy jest słowem?
wrap		0	zwiń ekran (połącz brzeg górny z dolnym i lewy z prawym)
xcor	op	0	współrzędna x żółwia
ycor	op	0	współrzędna y żółwia
+	op	2	dodawanie
–	op	1	mnożenie przez –1 lub odejmowanie
*	op	2	mnożenie
/	op	2	dzielenie
=	op	2	równość
<	op	2	mniejsze
>	op	2	większe
"	op	1	operator dosłowności
:	op	1	wartość
\	op	1	blokada interpretacji znaku
.bload		2	ładuj binarnie z pamięci zewnętrznej
.bsave		2	zapisz binarnie w pamięci zewnętrznej
.call		1	wywołaj binarnie
.contents		0	zawartość listy nazw procedur i danych
.deposit		2	włóż do pamięci pod podany adres
.examine		1	zobacz co jest w pamięci pod podanym adresem
.primitives		0	zawartość listy nazw procedur pierwotnych
.reserve		1	zajmij pamięć

.reserved	0	zajęte
.serialin	0	przyjmij przez wejście szeregowe
.serialout	1	wyślij przez wyjście szeregowe
.setserial	1	prędkość transmisji

UWAGA: komendami z kropką należy posługiwać się ostrożnie, gdyż nieuważne ich użycie może doprowadzić do nieprzewidzianych skutków, takich jak uszkodzenie potrzebnych zapisów w pamięci komputera, zablokowanie komputera przepełnieniem pamięci lub zablokowanie działania Logo.

Polskie Logo

Zestawienie polskich odpowiedników komend Sinclair Logo jest ułożone tak samo, jak opisy komend – nazwy angielskie są w porządku alfabetycznym. Składnia, sposób wykonania, zasady redagowania procedur są w obu wersjach takie same.

and	i
arccos	arccos
arccot	arcctg
arccotangent	arcctg
arcsin	arcsin
arctan	arctg
arctangent	arctg
ascii	ascii
back	wstecz
background	tło ↑
bf	bezpierw
bg	tło ↑
bk	ws
bl	bezost
bright	jaskrawo
butfirst	bezpierw
butlast	bezost
bye	dość
catalog	katalog
char	znak
clean	zmaż
clearscreen	czyść
cleartext	zmażtekst
copydef	powiel
copyscreen	drukobrazu
cos	cos
cosine	cos
cot	ctg
cotangent	ctg
count	długość
cs	cs
ct	zt
cursor	kursor ↑
define	określ
definedp	określone?
div	iloraz
dot	pkt
ed	red
edit	red
edns	redwn
emptyp	puste?
end	już
equalp	równe?
er	us
erall	usw
erase	usuń
erasefile	usuńplik
ern	usn
erns	uswn
erps	uswp
false	fałsz
fd	np
fence	pole
first	pierw
flash	migaj
forward	naprzód
fput	nap
heading	kąt ↑
hideturtle	sż
home	wróć
ht	sż
if	jeśli
int	entier, ent
inverse	negatyw
item	element
keyp	klawisz?
last	ost
left	lewo
list	lista
listp	lista?
load	ładuj
loadd	ładujr

loadscr	ładujo
lput	nak
lt	lw
make	przypisz, przyp
memberp	element?
name	nazwij
namep	jest?
nodes	wolne
normal	pozytyw
not	nie
numberp	liczba?
op	wy
or	lub
output	wynik
over	nadruk
pc	pisak ↑
pd	opu
pe	ścieranie
pencolour	pisak ↑
pendown	opu
penerase	ścieranie
penreverse	odwracanie
penup	podnieśpisak
po	po
poall	pow
pons	pown
pops	powp
pos	poz ↑
position	poz ↑
pots	potp
pr	pp
primitivep	pierwotne?
print	pisz
printoff	wyłączdruk
printon	włączdruk
product	iloczyn
pu	pod
px	odwracanie
random	losowa
rc	cz
readchar	czytajznak
readlist	czytajlistę
recycle	odśmieć
remainder	reszta
repeat	powtórz
right	prawo
rl	cl
round	zaokr
rt	pw
run	zrób
save	zapiszp
saveall	zapisz
saved	zapiszr
savescr	zapiszo
scrunch	proporcja ↑
se	zd
sentence	zdanie
setbg	tło
setborder	ramka
setbr	ramka
setcur	kursor
setcursor	kursor
setdrive	dysk
seth	kąt
setheading	kąt
setpc	pisak
setpos	poz
setscr	proporcja
setscrunch	proporcja
settc	koloryt
setx	xpoz
sety	ypoz
show	pokaż
shownp	widać
showturtle	pż
sin	sin
sine	sin
sound	nuta
sqrt	pierwiastek, pkw
st	pż
startrobot	startrobot
stop	stop
stoprobot	stoprobot
sum	suma
tan	tg
tangent	tg
tc	kt ↑
text	treść
textcolour	koloryt ↑
textscreen	teksty
thing	wartość, war
to	oto
toplevel	przerwij
towards	azymut
true	prawda
ts	ts
type	wpisz
wait	czekaj
window	okno
word	słowo
wordp	słowo?
wrap	sklej
xcor	xpoz ↑
ycor	ypoz ↑
.bload	.ładujbin
.bsave	.zapiszbin
.call	.wywołaj
.contents	.pamięć

.deposit	.umieść
.examine	.zobacz
.primitives	.pierwotne
.reserve	.zajmij
.reserved	.zajęte
.serialin	.przyjmij
.serialout	.wyślij
.setserial	.transmisja

Poza tym w polskim Logo są:

exp	funkcja wykładnicza
jas 1	jaskrawy ekran
jas 0	kasowanie jaskrawości ekranu
ln	logarytm naturalny
mig 1	miganie ekranu
mig 0	kasowanie migania ekranu
pi↑	liczba pi
zam	zamaluj, zamalowanie obszaru

Mając jakąkolwiek niepolską wersję Logo, w szczególności Sinclair Logo, można wprowadzić rozwiązanie zastępcze:

1. Zdefiniować w trybie graficznym wprowadzania z klawiatury (GRAPHICS, GRAPH na Spectrum +) wszystkie polskie litery ze znakami diakrytycznymi: ą, ę, ć, ń, ó, ś, ź, ż (oba jako graficzny odpowiednik z), ł. Definiowanie znaków jest opisane w podręcznikach Spectrum, do wpisywania danych do pamięci używa się . deposit.
2. Przedefiniować procedury języka, nadając im nowe nazwy (przy pomocy to ... end lub copydef). Nie da się w ten sposób przemianować procedur definiujących, to i end. Nie da się również zastąpić komunikatów angielskich polskimi. Takie przemianowanie procedur powoduje również dodatkowe zajęcie części pamięci na nowe nazwy, wskutek czego zmniejszy się nieco pamięć dostępna dla obliczeń. Dla obliczeń prostych i ćwiczeń z Logo na poziomie początkowym to wystarczy.

W treści

Cena zł 110,—

delta



ISBN 83-202-0522-0